ELECTRICAL ENGINEERING DEPARTMENT

UNIVERSITY OF TENNESSEE

KNOXVILLE TN 37916

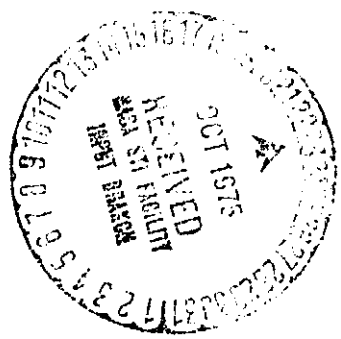
Gray-Level Transformations For Interactive Image Enhancement*

Principal Investigator: R. C. Gonzalez
Co-Investigator: B. A. Fittes

Final Report. Contract NAS8-29271

Technical Report TR-EE/CS-75-20

September, 1975



---

* This report was prepared by the Electrical Engineering Department, University of Tennessee, under Contract No. NAS8-29271 "The Application of Image Enhancement Techniques to Remote Manipulator Operation", for the George C. Marshall Space Flight Center of the National Aeronautics and Space Administration.

July 30, 1975

To the Graduate Council:

I am submitting herewith a thesis written by Barry Alan Fittes entitled "Gray-Level Transformations for Interactive Image Enhancement." I recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

R. C. Gonzalez

R. C. Gonzalez, Major Professor

We have read this thesis
and recommend its acceptance:

Robert E. Backenheimer

William R. Wade (Mathematics)

Accepted for the Council:

Vice Chancellor
Graduate Studies and Research

GRAY-LEVEL TRANSFORMATIONS FOR INTERACTIVE IMAGE ENHANCEMENT

A Thesis

Presented for the

Master of Science

Degree

The University of Tennessee

Barry Alan Fittes

August 1975

ACKNOWLEDGEMENTS

The author wishes to thank Dr. R. C. Gonzalez for his guidance in research and assistance in writing and proofreading this thesis. Appreciation is also expressed to Dr. R. E. Bodenheimer and Dr. W. R. Wade for serving on the thesis committee. Gratitude is extended to Diana Scott, whose assistance in typing this manuscript has been invaluable. Finally, the author wishes to thank his wife, Kay, for her patience, understanding, and encouragement throughout this project, as well as for proofreading the text of this thesis.

ABSTRACT

A gray-level transformation method suitable for interactive image enhancement is presented. It is shown that the well-known histogram equalization approach is a special case of this method. A technique for improving the uniformity of a histogram is also developed. Experimental results which illustrate the capabilities of both algorithms are described. Two proposals for implementing gray-level transformations in a real-time interactive image enhancement system are also presented.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE PAGE

FIGURE PAGE

CHAPTER I

INTRODUCTION

The area of research known as digital image processing can be broadly divided into two categories. The first category includes image processing for feature extraction and classification. The second category includes image restoration, enhancement and coding.

The first category deals with techniques such as edge and curve detection and texture measurement. These are operations which do not necessarily output a new image but rather detect features within an image that can be analyzed, for example, by some type of pattern recognition device.

Image coding is primarily concerned with the techniques used in transmission and storage of images in a digital format. In general, the objective of this type of work is to reduce the number of bits per picture element needed to accurately reproduce the image. In transmission, the objective is to reduce the bandwidth required in the communication channel. In the storage application of coding, the objective is to reduce the amount of bulk storage required for an image.

Image restoration is concerned with modifying an image that has been degraded in some way, in an attempt to arrive at an estimate of the original image. This degradation

could take the form of noise injected over a communications channel, motion blur or lens aberrations in the imaging system.

Image enhancement is an area of work concerned with manipulation of an image to extract details that might not be obvious in the original picture. Enhancement techniques can be broken down into two general classes: (1) transform domain methods, and (2) image domain methods. The first class includes techniques based on operations performed on a two-dimensional transform of an image such as, for example, the Fourier and Hadamard transforms. This type of processing is well-suited for removal of periodic noise, sharpening and smoothing operations.

The second class of enhancement techniques are those that operate on the image itself. These techniques can be further broken down into position-variant and position-invariant methods. The position-variant methods are those that view the image as a two-dimensional array of points. Hence, the operation performed on a point depends on its position in the picture. Position-invariant methods of enhancement are the subject of this work. These are the methods that operate directly on a picture element, without regard to its position in the picture. One class of these methods is known as gray-level mapping. The objective of this approach is to find a function that will map the gray levels of an image into another set of levels to enhance some feature of

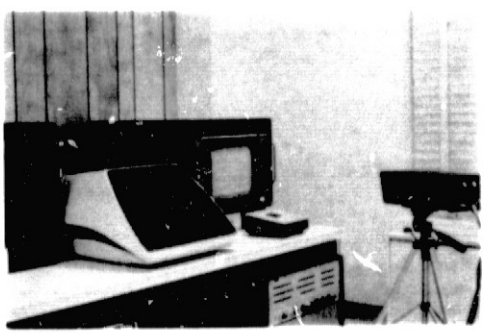
the image. As will be seen in the following chapters, this approach readily lends itself to implementation in an interactive image enhancement system which requires no knowledge of enhancement techniques on the part of the operator.

A brief review of the terminology and image formats used in this report will be useful in establishing the notation used in subsequent chapters. A picture element will be referred to as a pixel. It is a single point in the picture and can assume any one of a number of quantized values. In the system used for this work, each pixel is represented by eight bits or 256 discrete values. The values can be considered to span the range [0,255] or they can be normalized to a range of [0,1]. At times it may also be advantageous to view the range of pixel values as [-0.5,0.5]. The representation used depends on the particular application.

The system used in the experiments to be described in subsequent chapters is shown in Figure 1. It consists of a Digital Equipment Corporation (D.E.C.) PDP 11/40 minicomputer with thirty-two thousand words of core memory. Standard peripherals interfaced to the system include two high density disk drives, an industry-compatible magnetic tape unit, an alphanumeric display terminal, a high speed paper tape reader/punch, a card reader, and a line printer.

The image-inputting device consists of a standard closed-circuit television camera and a video sampling system. The sampling system consists of a hardware unit that monitors

(a)



(b)

Figure 1. Image processing laboratory at the University of Tennessee.

the horizontal and vertical synchronization signals from the television camera to generate a series of pulses that initialize an analog to digital converter to sample the video signal.[1]* It also consists of the software to drive the system.[2]

The video display system used for outputting the processed results consists of a Model 6600 Display System manufactured by Data Disc, Inc. This system generates a video signal with 480 by 640 individually addressable points. For the purposes of this work, the pictures were displayed in a 480 by 512 format. The video display unit as it is now configured generates a 256 level monochrome signal, and a 256 level color signal that drive standard black and white and color television monitors.

The sampling unit and all of the software developed thus far are capable of handling three different picture sizes: 128 x 128, 256 x 256, and 512 x 512. These will be referred to as small, medium and large, or S, M, and L.

Previous work done on gray level transformations will be discussed in the following chapters, as well as the algorithms that have been developed during the course of this investigation. The considerations relative to the interactive nature of the processing will also be discussed.

---

*Superscript numbers in parentheses refer to those entries listed in the List of References.

Finally, experimental results will be discussed and conclusions will be drawn as to the value of this work and what might lie in the future for this type of image processing.

CHAPTER II

GRAY-LEVEL TRANSFORMATIONS

Background

The objective of the work in gray-level transformations is to find a function that will modify the gray levels of an image, in order to enhance some feature of the image. Letting x represent the gray levels of an image to be enhanced, the transformations of interest are of the form

$$y = T(x) \tag{1}$$

which map the levels x into corresponding levels y to obtain an image which has been enhanced in some manner. It is assumed in the following discussion that the levels have been normalized to the interval [0,1], where zero represents black and one represents white in the gray scale.

Two conditions must be imposed on the transformation function T(x) to obtain meaningful results. First, T(x) must be monotonically increasing in the sense that

$$T(x_1) \leq T(x_2) \qquad (x_1 < x_2). \tag{2}$$

The implication of this constraint is that if pixel x_1 is lighter than pixel x_2 in the original image, then pixel x_1 will also be lighter than pixel x_2 in the enhanced image.

If this condition is not met, the resulting picture could be meaningless.

The second condition imposed on the transformation is that

$$0 \leq T(x) \leq 1 \qquad (0 \leq x \leq 1). \tag{3}$$

This serves to maintain a mapping that will be consistent with the allowed range of gray levels.

The variable x is a random variable. For each image to be enhanced, x has a probability density function or p.d.f. denoted $p_x(x)$. In the enhanced image, the levels will be denoted by y with p.d.f. $p_y(y)$. For a discrete or digitized image, the p.d.f. is a discrete function known as a histogram which specifies the number of occurrences of each gray level within an image. For simplicity, the following development is presented in continuous mathematics. The discrete equivalents can be obtained without difficulty.

Given a transformation function T(x) and a density $p_x(x)$, statistical theory provides a means for estimating the p.d.f. of the random variable, y, given that y = T(x). The p.d.f. of the variable y in Equation (1), page 7, can be obtained using the relation

$$p_y(y_i) = \frac{p_x(x_1)}{|T'(x_1)|} + \ldots + \frac{p_x(x_n)}{|T'(x_n)|} \tag{4}$$

where $y_i = T(x_j)$; $j = 1, n$; and $T'(x)$ denotes the derivative of T with respect to x.[3] Since one of the constraints on the transformation T is that it be strictly increasing, there will be only one solution to Equation (1), page 7. In addition, the absolute value sign may be dropped since $T'(x)$ will always be positive because of this condition. Hence, Equation (4), page 8, becomes

$$p_y(y) = \frac{p_x(x)}{T'(x)} = \frac{p_x(x)}{\frac{dy}{dx}} = p_x(x)\frac{dx}{dy} \tag{5}$$

where $x = T^{-1}(y)$. Thus, the p.d.f. of the enhanced image can be estimated knowing only the original p.d.f. and the transformation function.

It is noted that knowledge of the function $p_x(x)$ and $p_y(y)$ yields information about the physical characteristics of the images. For instance, the average brightness, or mean, of the original image is given by

$$\bar{x} = \int_0^1 x\, p_x(x)dx, \tag{6}$$

while the variance,

$$\sigma_x^2 = \int_0^1 (x-\bar{x})^2 p_x(x)dx, \tag{7}$$

is a measure of the amount of contrast in the image. The brightness and contrast parameters are of fundamental importance in describing the characteristics of an image. They

also provide a basis for comparing images mathematically using parameters that can be interpreted visually.

Histogram Equalization

A very popular approach to the enhancement problem is to generate a uniform p.d.f. The objective in this case is to redistribute the gray levels in a picture such that each gray level within the p.d.f. contains the same number of pixels. The intention is to optimize the use of the available levels. This technique is known as histogram equalization or distribution linerization.[4, 5, 6]

The transformation used is of the form

$$y = T(x) = \int_0^x p_x(s)ds \qquad (0 \leq x \leq 1), \qquad (8)$$

where s is a dummy variable of integration. This function is known as the cumulative distribution function or c.d.f. of x. From Equation (8), it follows that

$$\frac{dx}{dy} = \frac{1}{p_x(x)} , \qquad (9)$$

therefore, Equation (5), page 9, reduces to

$$p_y(y) = 1 \qquad (0 \leq y \leq 1) \qquad (10)$$

which is the desired uniform density.

A heuristic argument can be presented in favor of histogram equalization. It is not unreasonable to assume that the quality of an image will be generally improved if its

pixels are evenly distributed over the entire available gray scale. All of the significant details present in the original image are generally preserved, while the subtle details not obvious in the original image are enhanced. This can best be illustrated by example.

Figure 2 is the image of a woman's face with a shadow cast over one side. Figure 3(a) illustrates the histogram of the image before processing. The contrast of the image is rather poor, as would be expected from the narrow gray-level range spanned by its histogram. The c.d.f. of this histogram appears in Figure 4. The reference line is to compare the actual c.d.f. with the c.d.f. of a uniform density which is the curve $y = x$.

The image of Figure 2 was transformed using the curve in Figure 4. The result appears in Figure 5, with its corresponding histogram in Figure 3(b). The transformation has tended to flatten and expand the histogram. The gaps in the resulting histogram are due to the fact that the pixels assume only discrete values. The c.d.f. of the resulting histogram appears in Figure 6. Note that it more closely approximates the reference line. The histogram indicates that the contrast of the image has been increased and Figure 5 confirms this point. Note that there is considerably more detail visible in the enhanced image than in the original. For most images, this has proven to be the case.

Figure 2. Original image of a woman's face with sidelighting.

Figure 3. Histogram of Figure 2 (a) before processing and (b) after equalization.

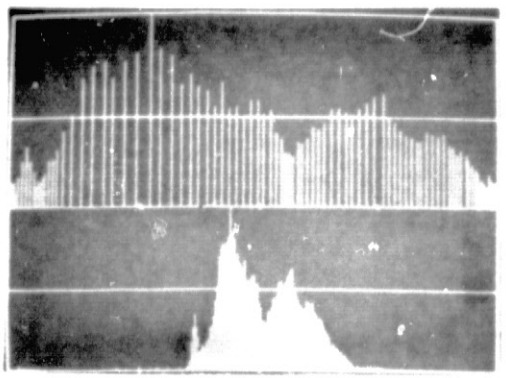
Figure 4. Cumulative distribution function for Figure 2.

Figure 5. Histogram-equalized version of Figure 2.

Figure 6. Cumulative distribution function for Figure 5.

Figure 2

(b)



(a)

Figure 3

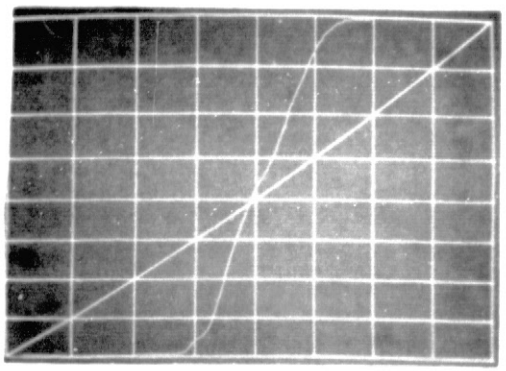


Figure 4

Figure 5

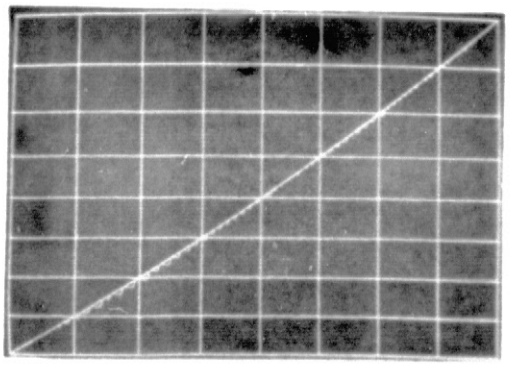


Figure 6

Contrast Enhancement Methods

Another type of gray level mapping is referred to as contrast enhancement or contrast stretching.[5, 7] The objective of this type of processing is to manipulate the spread of one or more areas of the density function. Some example transformation curves appear in Figure 7. Figure 7 (a) shows a transformation function that could be used on an image where all or most of the pixels have values on the interval [a,b]. This transformation then spreads or stretches the density to cover the entire [0,1] range. Figure 7(b) shows a curve that could be used to spread the dark region of a picture, while Figure 7(c) illustrates a curve that will operate similarly on the light areas of a picture. Finally, Figure 7(d) shows a curve that would tend to spread out the middle range of a picture.

One algorithm has been proposed that generates a function similar to that in Figure 7(d).[8] The function used is of the form

$$y = T(x) = f^{-1}(\tan^{-1}(c*\tan(f(x)))) \qquad (x \neq 0 \text{ or } 1), \quad (11)$$

and

$$T(0) = 0, \; T(1) = 1, \quad (12)$$

where f is a function that maps the pixel spectrum [0,1] onto the domain of the tangent function $[-\pi/2, \pi/2]$. The constant c determines the degree of enhancement.

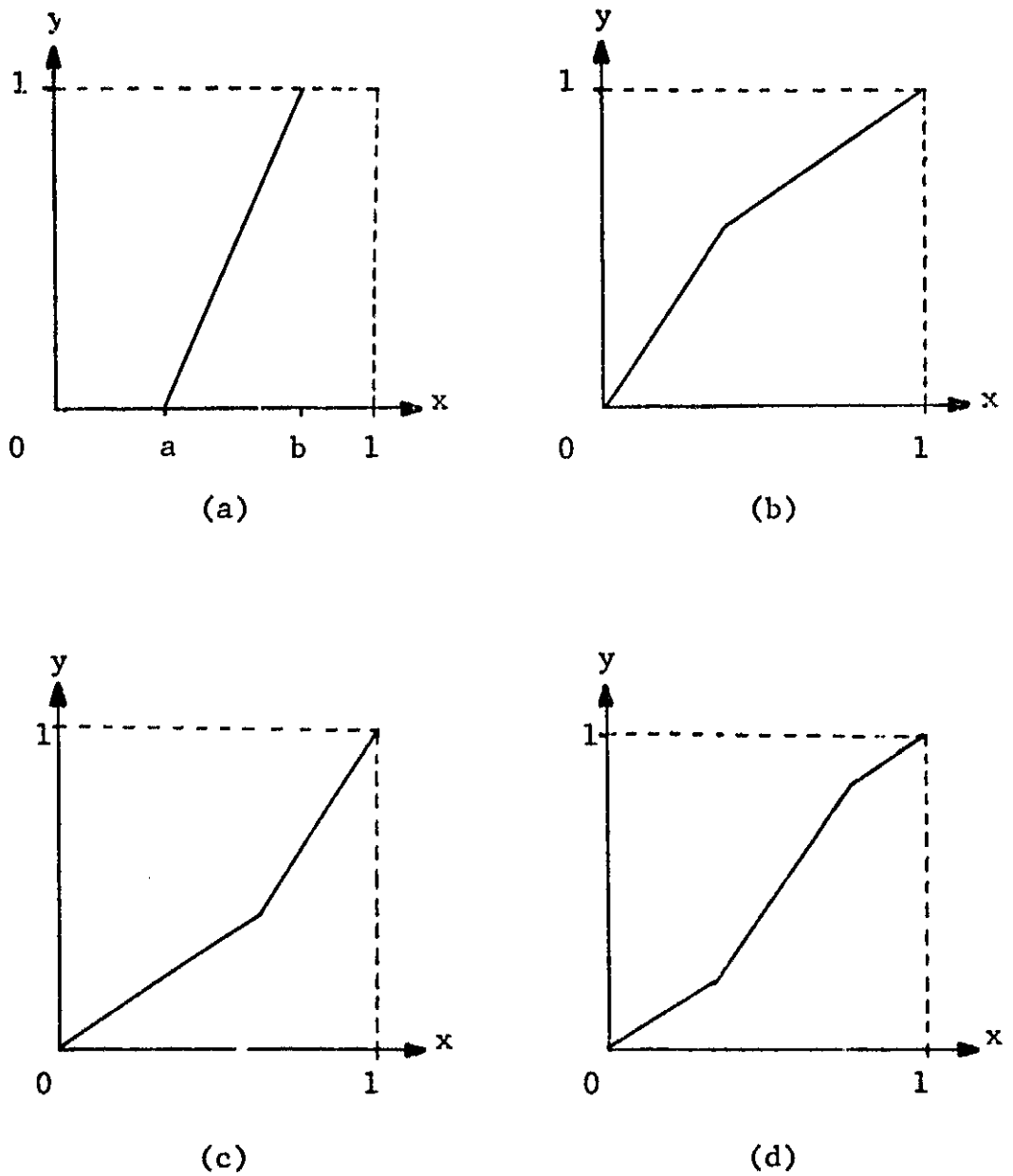
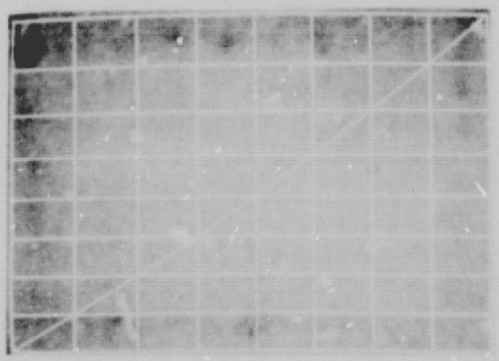


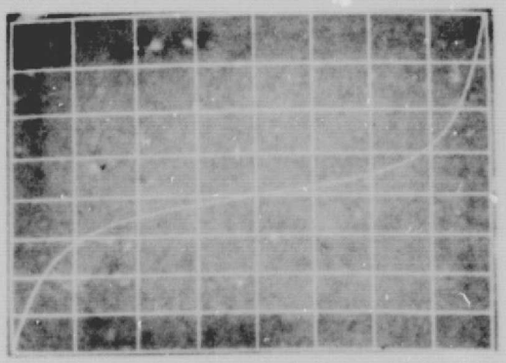
Figure 7. Transformation functions for (a) increasing the dynamic range of a picture, (b) spreading the dark area, (c) spreading the light area, and (d) spreading the middle range.

The tangent function maps its domain onto the infinite real line. The positive constant c multiplies the points on the real line to either spread them out or compress them. Then the inverse tangent and inverse f are taken to map the points back to the interval [0,1]. Hence, the value of the constant c directly controls the variance or spread of the density function of the enhanced image.

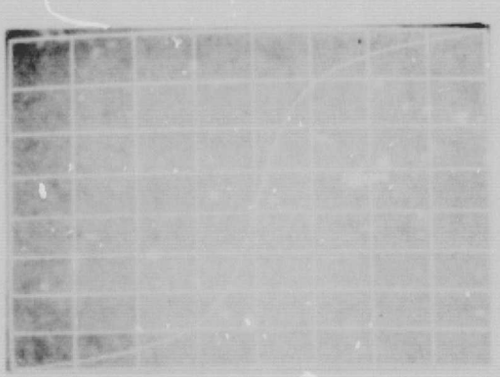
The function f, as it was proposed, maps the mean of the density to the mid-point of the interval $[-\pi/2,\pi/2]$ which is zero. Similarly, the inverse f maps the mid-point back to the mean of the original image as explained in Reference 8. The inverse of f can be replaced by another function similar in nature to the inverse of f. This function can be used to map the mid-point of the tangent domain to a mean specified by an operator, rather than the original mean. This gives the user control over the mean as well as the variance of the result. For the following examples, it is assumed that the old mean and the new mean are both 0.5. Figure 8(a) is the transformation function defined in Equation (11), page 14, that was generated with c = 1.0, which yields the original image. Figure 8(b) was generated with c = 0.2. Assuming that most of the pixel values lie in the mid-range of the spectrum, this transformation will tend to decrease the contrast of the picture. Figure 8(c) shows the function that was generated with c = 5.0. This transformation will tend to increase the contrast of the picture. Note



(a)



(b)



(c)

Figure 8. Examples of transformation function in Equation (11) with (a) c=1.0, (b) c=0.2, and (c) c=5.0.

that each of these curves has odd symmetry about the point (0.5,0.5). This is due to the fact that the tangent function has odd symmetry about the origin.

Figure 9 is the result of contrast enhancement as proposed in Reference 8. The original of this image appears in Figure 2, page 12. Fairly good results have been obtained with this algorithm in most cases. This approach gives the user direct control over the contrast of a picture through the parameter c, and the brightness through the function f.

Drawbacks of Previous Methods

Two methods of gray-level transformations have been presented with examples to show their effectiveness. Arguments are given below to show the inadequacies of each method in terms of implementation in an interactive environment and in terms of the algorithm itself.

The major drawback of histogram equalization is that the algorithm is fixed and, therefore, the user has no control over the process. It will be shown below, however, that histogram equalization can be used as an intermediate step in an algorithm that provides results superior to either of the methods described previously. Further, it will be shown that histogram equalization is a special case of this method.

The contrast enhancement approach proposed in Reference 8 lends itself well to implementation in an interactive environment, requiring only two parameters to control it.

Figure 9. Contrast enhanced version of Figure 2.

The algorithm is fairly straight-forward and does not involve much computation. However, it has one major flaw. As pointed out above, the function as defined in Equation (11), page 14, is symmetric about the mean of the original image as was illustrated in Figures 8(b) and 8(c), page 17. This implies that the density of the image to be enhanced should be symmetric about its mean, a condition which is seldom satisfied in practice. This algorithm is certainly suitable for manipulating the brightness and contrast of pictures, however, this is the extent of its usefulness. It lacks the flexibility required to manipulate an unsymmetric density in an unsymmetric manner.

Direct Histogram Specification

The gray-level mapping method developed below, like those presented above, is based on transforming the probability density function of an image to be enhanced. The process is different, however, in that a transformation is developed that will produce an image with a density function $p_y(y)$ which has been interactively specified by an operator.

Using Equation (8), page 10, the gray levels x of an image can be transformed to a new set of levels z, that is,

$$z = H(x) = \int_0^x p_x(s)ds \qquad (0 \leq x \leq 1). \qquad (13)$$

From the above discussion it is known that $p_z(z)$ will be a uniform density function. If the inverse of Equation (13),

$H^{-1}(z)$ is applied to the new density, the original image with density $p_x(x)$ is obtained. It is noted, however, that if Equation (8), page 10, is applied to a different density $p_y(y)$, the result will be

$$z = G(y) = \int_0^y p_y(s)ds \qquad (0 \leq y \leq 1). \qquad (14)$$

Although G(y) is in general different from H(x), $p_z(z)$ is the same if either Equation (13), page 20, or Equation (14) is used. If the inverse function $G^{-1}(z)$ is applied to the z levels, the result will be a set of y levels with the specified density $p_y(y)$. In other words, if the z levels have a uniform density, the desired $p_y(y)$ can be obtained using the inverse mapping $G^{-1}(z)$. It is noted that both H(x) and $G^{-1}(z)$ must satisfy the condition of monotonicity as set forth in Equation (2), page 7. Therefore, if the original image is first histogram-equalized, and the new (uniform) levels are inverse-mapped using the function $G^{-1}(z)$, the result will be an image whose gray levels have the desired density $p_y(y)$.

The procedure is illustrated in Figure 10. Figure 10(a) shows the original density function $p_x(x)$ which is computed from the image to be enhanced. Figure 10(b) shows the cumulative distribution function H(x) which is used to map the variable x into the variable z. Figure 10(c) illustrates the result of this transformation. Figure 10(d) is the c.d.f. of the desired density, G(y), the inverse of which is used to map the variable z back into the desired density $p_y(y)$.

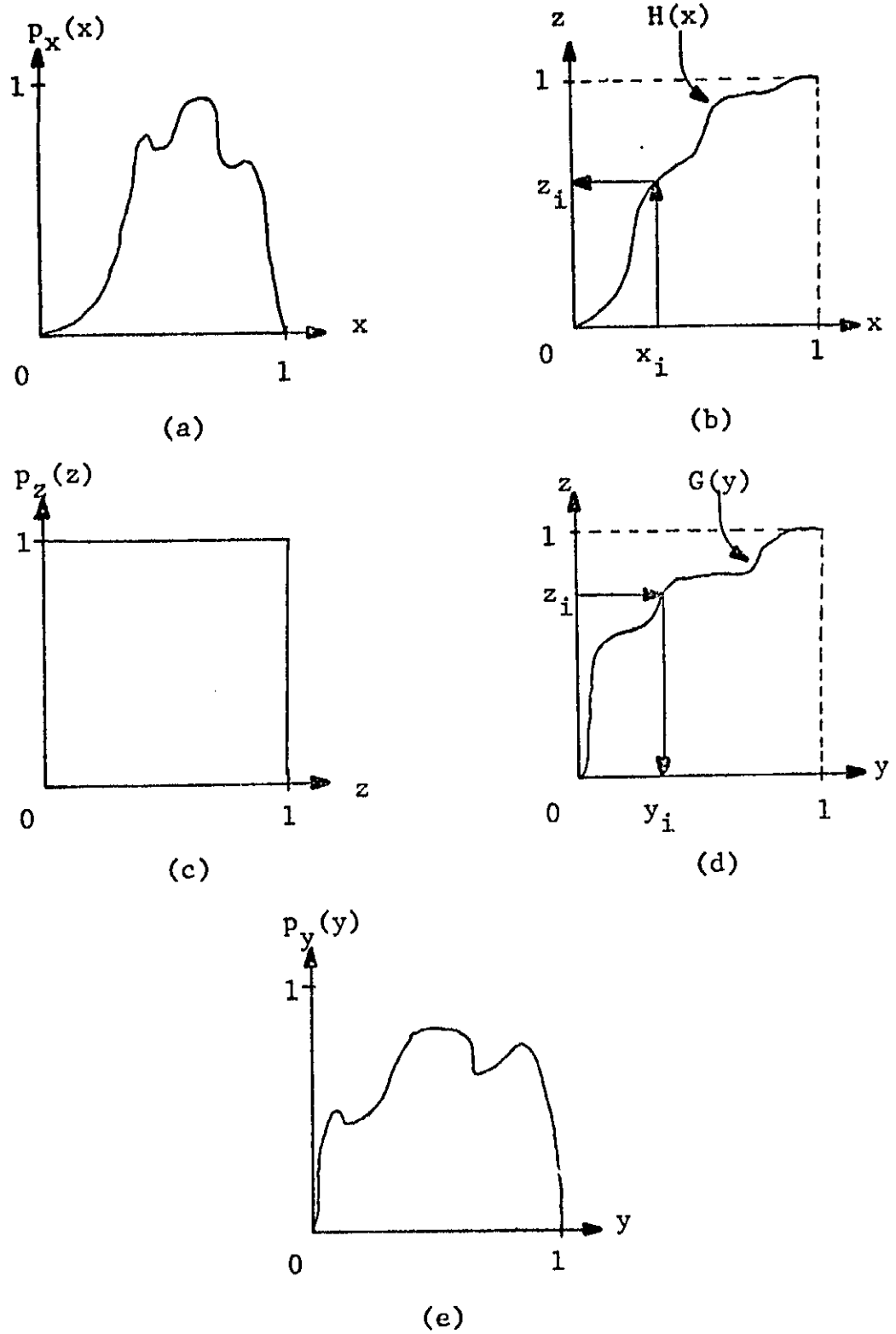


Figure 10. Steps in the direct histogram specification transformation procedure.

Figure 10(e) illustrates the result of this transformation which is the desired density.

This procedure can be expressed in the form of an equation that specifies the transformation from the given density $p_x(x)$ to the desired density $p_y(y)$. The relation is given by

$$y = G^{-1}[H(x)] \tag{15}$$

where H(x) is the c.d.f. of x and G(y) is the c.d.f. of y. It is noted that histogram equalization is a special case of this technique obtained by letting $G^{-1}[H(x)] = H(x)$.[9]

This method of transformation from one density to another is well known to statisticians. The problem in using this technique with continuous functions is that for most functions, the inverse of the c.d.f. is difficult if not impossible to evaluate analytically. For discrete functions, however, the inverse can be approximated without difficulty. Since the emphasis of this investigation is on discrete images, it follows that Equation (15) can be used as a tool for image enhancement, provided that a desired histogram has been specified. As will be shown in the following chapter, the problem of specifying a histogram can be formulated in an interactive manner, thus giving the operator control over the enhancement process.

As an example of the superiority of this method over histogram equalization, consider Figure 11 which shows the image of a quarter taken inside a poorly illuminated box.

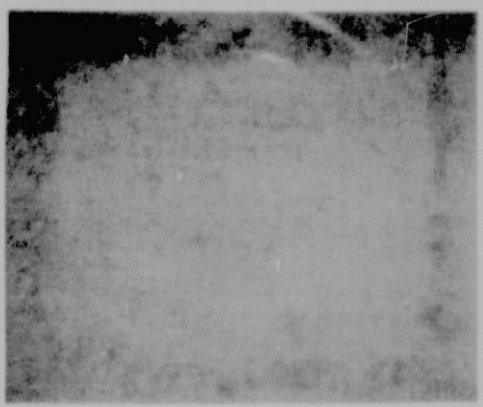


Figure 11. Original image of a quarter in a poorly illuminated box.

Figure 12 is the result of histogram equalization. Figure 13 illustrates the result of contrast enhancement as proposed in Reference 8. Figure 14 is the result of processing with a method that uses the transformation given in Equation (15), page 23. The method used to specify the resultant density will be described in Chapter III. Note that Figure 14 provides much more information about the object in the image than do Figures 12 or 13. The histograms for Figures 11, 12, 13, and 14 appear in Figures 15(a), 15(b), 16(a), and 16(b), respectively. It is noted that the histograms of the three enhanced pictures do not differ greatly; however, the amount of visual information in the picture does.

Improving the Uniform Density

The success of the above algorithm in approximating a specified density relies on the fact that a uniform histogram is achieved in the process of histogram equalization. Due to the fact that the pixels assume only discrete values, however, the result of the equalization process is generally only an approximation to a uniform histogram. Figures 3(a) and 3(b), page 12, illustrate a good example of the non-uniformity that is achieved in practice. If the original density has even fewer gray levels present, the approximation gets worse as is illustrated in Figures 15(a) and 15(b). The root mean square (R.M.S.) error between each of these equalized histograms and a true uniform histogram is .0063 and .031, respectively.

Figure 12. Histogram-equalized version of Figure 11.

Figure 13. Contrast enhanced version of Figure 11.

Figure 14. Image of Figure 11 after enhancement by histogram specification.

Figure 15. Histograms of (a) Figure 11 and (b) Figure 12.

Figure 16. Histograms of (a) Figure 13 and (b) Figure 14.

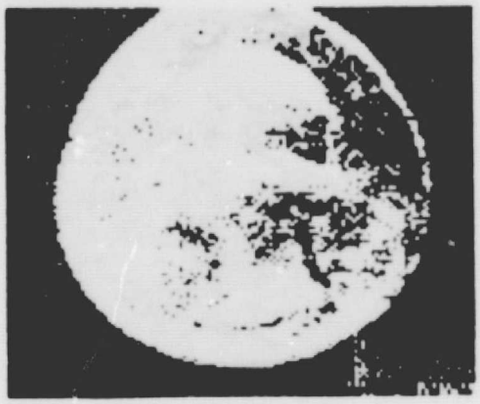
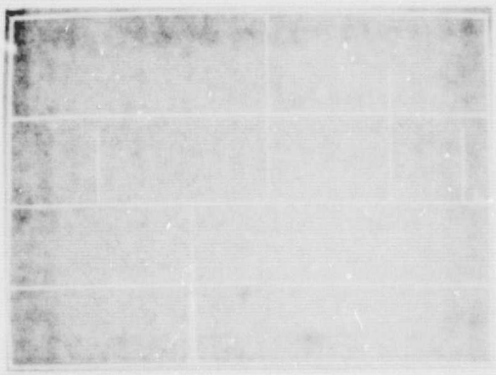


Figure 12

Figure 13

Figure 14

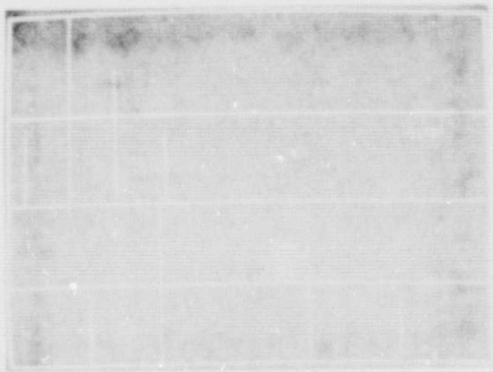
(b)



(a)

Figure 15

(b)



(a)

Figure 16

Due to the discrete nature of the pixel values, even if a uniform histogram is obtained, it will not necessarily yield exactly the histogram that was specified. Theoretically, however, this is the best that can be achieved. Hence, a uniform histogram can be used as a reference in determining how close the specified and resultant histograms actually are. For example, Figure 17 illustrates a histogram that was specified using the method to be outlined in Chapter III. Figure 18 shows the result of mapping a perfect, uniform histogram to the histogram shown in Figure 17. The R.M.S. error between the specified and resultant histograms was computed to be .0017. Figure 19 illustrates the result of mapping the histogram of the image in Figure 2, page 12, to the histogram specified in Figure 17. The R.M.S. error was computed to be .006 which is a 243.8 percent increase over the error obtained with the uniform histogram.

An algorithm has been developed that can be used to preprocess an image in order to improve the uniformity of the histogram resulting from the equalization procedure. The algorithm was adapted from some results in texture analysis.[10] This algorithm differs from those previously described in that it considers a pixel and a small neighborhood around it, rather than just the pixel itself, when determining the new value of the pixel.

The image is assumed to consist of a two-dimensional array A with elements A(i,j). A new image B is computed

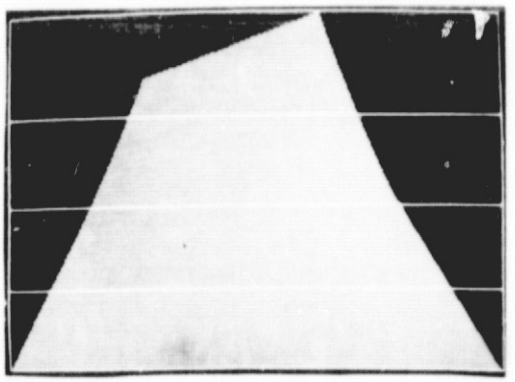


Figure 17. Example of a histogram specified by a user.

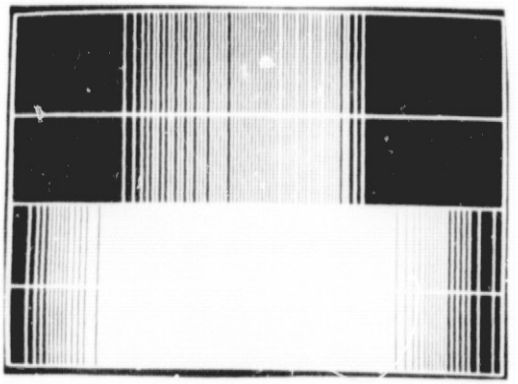


Figure 18. Mapping of a uniform histogram to that in Figure 17.

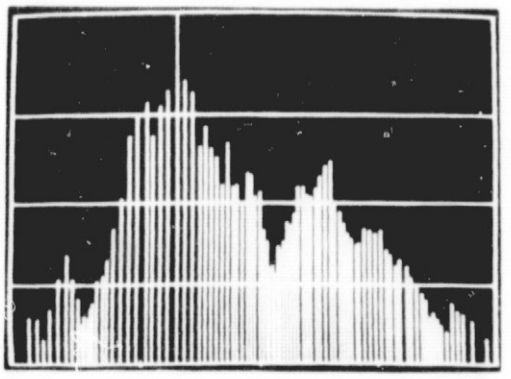


Figure 19. Mapping of the image in Figure 2 to the histogram in Figure 17.

from A using the following relation:

$$B(i,j) = \sum_{m=-1}^{1} \sum_{n=-1}^{1} A(i-m,j-n) \qquad (16)$$

assuming that the pixel values are on the interval [0,255]. This yields an image with a resolution of one part in 2,304 rather than one part in 256. This is effectively an increase from eight bits to approximately 11.2 bits. This new image B with a 2,304 point density function can then be histogram-equalized and quantized back into 256 levels.

The significance of this algorithm is that it can differentiate between two different pixels with the same value that appear in the image in different contexts. For example, suppose that pixels x_1 and x_2 have the same value, but pixel x_1 is situated in a very dark region of the picture while pixel x_2 is in a lighter region. This algorithm, rather than map both x_1 and x_2 to the same value, will map x_1 to a darker value than x_2.

The reason that histogram equalization does not yield a uniform density is that for any gray level, all of the pixels with that value will be mapped to the same new value. This algorithm, however, provides a way of mapping contextually different points with the same value to different values. This allows large groups of points with the same value to be broken down into smaller groups with different values in a meaningful manner.

One of the advantages of this preprocessing technique is that it tends to average out the noise in an image. Another advantage is that successive iterations through the algorithm tend to produce better approximations to the desired uniform histogram. Hence, the histogram obtained in the second step of the algorithm will more closely approximate the histogram that was specified. Figure 20 illustrates the histogram of Figure 2, page 12, after one iteration through the preprocessing algorithm. The R.M.S. error of this histogram with respect to a uniform histogram is .00084. The use of this preprocessed image when input to the histogram specification algorithm results in the histogram shown in Figure 21. The R.M.S. error for this histogram was computed to be .00195, which is only a 12.2 percent increase over the reference. Figures 22 and 23 illustrate a second iteration through the preprocessor. The R.M.S. error of Figure 22 with respect to the uniform histogram in this case has decreased to .00029. The error in the resultant histogram has decreased to .00186, which is only 7.0 percent above the reference. Note that both iterations have reduced the R.M.S. error at both the first and second steps of the specification algorithm. Experience has shown that after two iterations the histogram does not improve significantly.

There is a problem with the preprocessing method described above. Like neighborhood averaging, it tends to blur the picture. Figure 24, for example, illustrates a picture

Figure 20. Histogram of the image in Figure 2 after one iteration through the preprocessor.

Figure 21. Mapping of the histogram in Figure 20 to the histogram in Figure 17.

Figure 22. Histogram of the image in Figure 2 after two iterations through the preprocessor.

Figure 23. Mapping of the histogram in Figure 22 to the histogram in Figure 17.

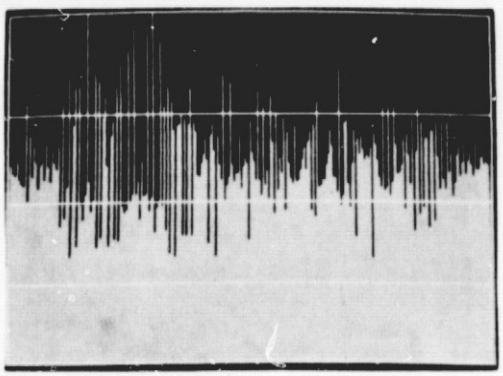


Figure 20

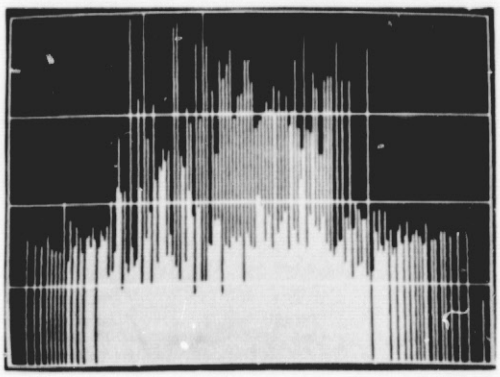


Figure 21

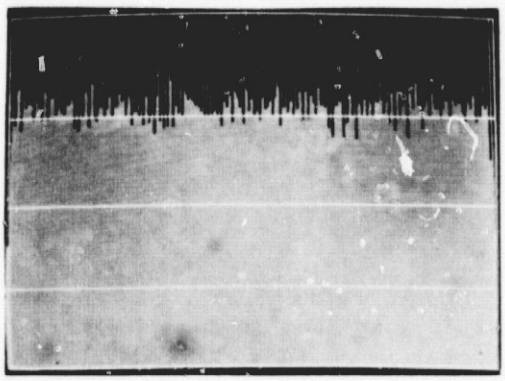


Figure 22

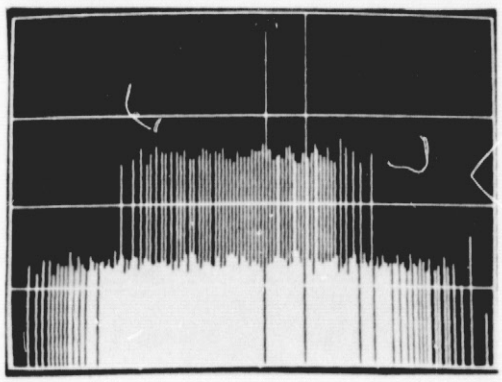


Figure 23

Figure 24. Original image of a woman's profile.

before processing of any type. Figure 25 shows the same picture after histogram equalization. Figures 26 and 27 show the results of one and two iterations of preprocessing, respectively. Note that the detail has been blurred to some extent.

A rather heuristic method has been developed for eliminating the blur without significantly affecting the general shape of the histogram. Let C(i,j) represent nine times the value of A(i,j). (It takes nine points from A to yield one point in B.) After the point B(i,j) is computed as in Equation (16), page 30, the difference between this point and C(i,j) is computed. A histogram of the absolute value of these differences is then computed. The points where the difference is zero are disregarded. Since the reason that the picture blurs is that the computed value of B(i,j) differs too much from C(i,j), it was decided to use the mean of this histogram as a threshold on either side of the point C(i,j). If B(i,j) lies within these limits, the point is not changed. If it lies outside of these limits, then the point is reassigned a value inside the limits. An attempt is made to distribute these values uniformly throughout the interval so as to aid in achieving a uniform density.

Figures 28 and 29 illustrate the result of processing Figure 24, page 34, through one and two iterations of the preprocessor with thresholding. Note that there is no loss of detail with respect to the histogram-equalized image of

Figure 25. Histogram-equalized version of Figure 24.

Figure 26. Image of Figure 24 after one iteration of the preprocessor.

Figure 27. Image of Figure 24 after two iterations of the preprocessor.

Figure 28. Image of Figure 24 after one iteration of the preprocessor with thresholding.

Figure 29. Image of Figure 24 after two iterations of the preprocessor with thresholding.

Figure 25

Figure 26

Figure 27

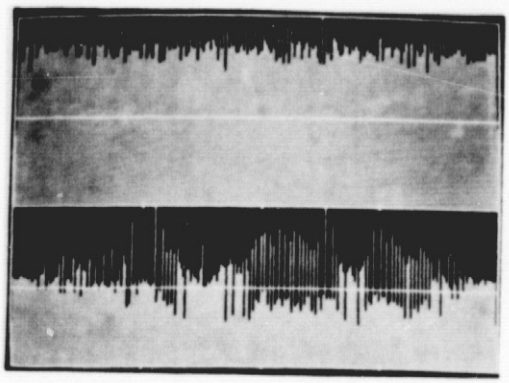
Figure 28

Figure 29

Figure 25, page 36. Figures 30(a), 30(b), 31(a), and 31(b) illustrate the histograms of Figures 26, 27, 28, and 29, page 36. The R.M.S. errors were calculated to be .001, .00028, .00095, and .00048. Note that processing with and without thresholding makes very little difference in the histograms that were generated. It is obvious, however, that there is a very distinct difference in the quality of the images.

In summary, an algorithm has been developed for obtaining a specified histogram from an arbitrary histogram using histogram equalization as an intermediate step. In general, the accuracy of the algorithm in achieving the desired histogram depends on the success of the histogram equalization in obtaining the desired uniformity. Hence, a preprocessing technique has been implemented that acts to modify the image such that a better approximation to a uniform histogram can be achieved. Further results using this enhancement approach are presented in Chapter IV.

(b)



(a)

Figure 30. Histograms of (a) Figure 26 and (b) Figure 27.

(b)

(a)

Figure 31. Histograms of (a) Figure 28 and (b) Figure 29.

CHAPTER III

INTERACTIVE SPECIFICATION OF HISTOGRAMS

As was stated in Chapters I and II, the principal objective of this investigation was to develop an algorithm that could be implemented interactively. The reasoning behind this objective is that the quality of a picture is difficult to quantify. The computer can manipulate an image, but only a human with an intuitive knowledge of what is required can make the decision as to whether or not the computer has produced a useful result.

With the enhancement algorithm developed in Chapter II, all that is required is the specification of a desired histogram. The algorithm then computes a transformation function based on the histogram specified by the user and the histogram of the original picture. There are two basic categories of users that must be considered when devising a method for specifying histograms. The first category includes users who are familiar with statistical theory and image processing techniques and are interested in the theoretical aspects of the algorithm. The second category includes users who are only interested in the final results of the enhancement system. Both categories of users are considered in the following pages.

Very sophisticated schemes for specifying histograms can be developed when it is known that the user has some degree of expertise in the field. One of the simplest methods would be to specify a histogram that approximates a normal density which requires only two parameters to define it. More complex schemes could include piece-wise linear histograms with many segments. A procedure can be implemented which will allow the user to experiment and determine what types of histograms produce certain features in an image.

One of the more interesting methods of specifying a histogram involves merely sketching the desired curve. This curve can then be input to the computer in several different ways. A light pen could be used in conjunction with a graphics terminal for such an application. If a digitizing television camera is available, the curve can be sketched on paper and then digitized using some type of edge-following algorithm. Another method would be to use a joystick or trackball to move a cursor along the curve and digitize the X-Y position of the cursor. An example using this type of histogram input will be presented in Chapter IV. A fourth alternative would be to use a graphics tablet to sketch the curve.

The possibilities for this type of specification are unlimited. The advantage is that the user can specify any shape of histogram that is required, without being limited

by the structure of a histogram specification algorithm. From this standpoint, the user can learn what general shape of histogram yields a particular characteristic in a picture. An algorithm that simplifies the specification problem can then be formulated from this information.

The methods mentioned above offer a wide range of options in the histogram specification process. The drawback is that they require knowledge of statistics and image processing techniques in order to be effectively used. The untrained user may find it very difficult to extract usable output from such a system. A method of specifying a curve that is relatively simple, but effective, must be developed for this class of user.

The first step in devising an algorithm that generates histograms is to determine the number of parameters that are required to adequately specify the major characteristics of a histogram. Two parameters that are obviously required are the mean and variance. As stated previously, the mean determines the average brightness level of the picture, while the variance is a measure of the contrast of the picture. In terms of the histogram itself, the variance is a measure of the spread of the histogram about the mean.

In the terminology of statistics, the mean and variance are referred to as the first moment about the origin and the second moment about the mean, respectively. They are the only parameters required to completely specify the familiar

normal density. However, the normal density is symmetric about the mean and as stated in Chapter II, a symmetric histogram is a rare case in practice. Something is needed to quantify the asymmetry of the histogram.

The third moment about the mean of the histogram can be effectively used to express the asymmetry of a histogram. The third moment is given by the following relation:

$$m_3 = \int_0^1 (x-\bar{x})^3 \, p_x(x)\,dx. \tag{17}$$

There are several ways to measure the asymmetry or skewness of the histogram. One is called the moment coefficient of skewness and is given by the relation

$$a_3 = \frac{m_3}{(\sigma_x^2)3/2}, \tag{18}$$

where σ_x^2 is the variance as defined in Equation (7), page 9. This coefficient will be used throughout the following discussion. There are two other coefficients of skewness known as Pearson's first and second coefficient of skewness.(11, 12) They are defined as follows:

$$Sk_1 = \frac{3(\bar{x}-Mo)}{\sigma_x} \tag{19}$$

where Mo is the mode of the histogram and

$$Sk_2 = \frac{3(\bar{x}-Md)}{\sigma_x} \tag{20}$$

where Md is the median of the histogram.

For perfectly symmetrical histograms, all three of these coefficients are zero. If the coefficient is negative, the histogram is said to have negative skewness or to be skewed to the left. This occurs when the histogram has a longer tail to the left of the central maximum than to the right, as in Figure 32(a). Figure 32(b) shows a histogram that is skewed to the right. In this case the skewness coefficient is positive.

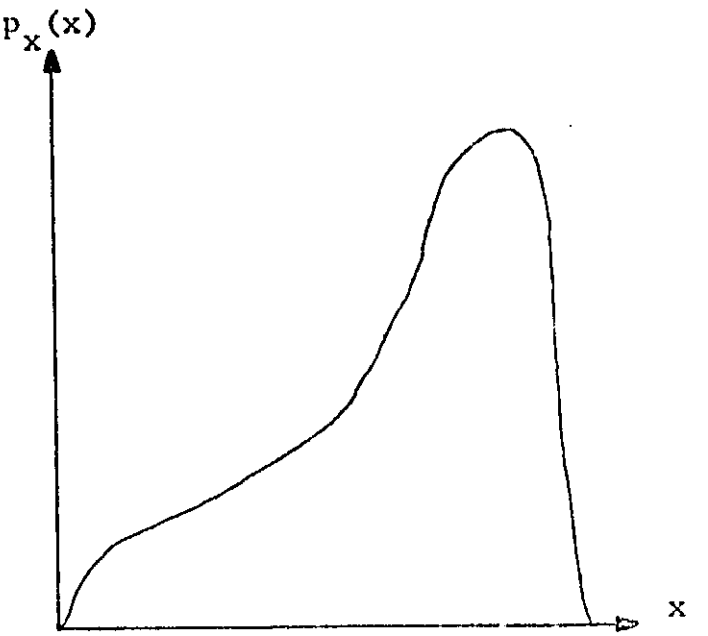
Most image histograms found in practice can be adequately described statistically with the addition of the fourth moment about the mean (also called kurtosis), which is given by the relation

$$m_4 = \int_0^1 (x-\bar{x})^4 \, p_x(x)dx. \tag{21}$$

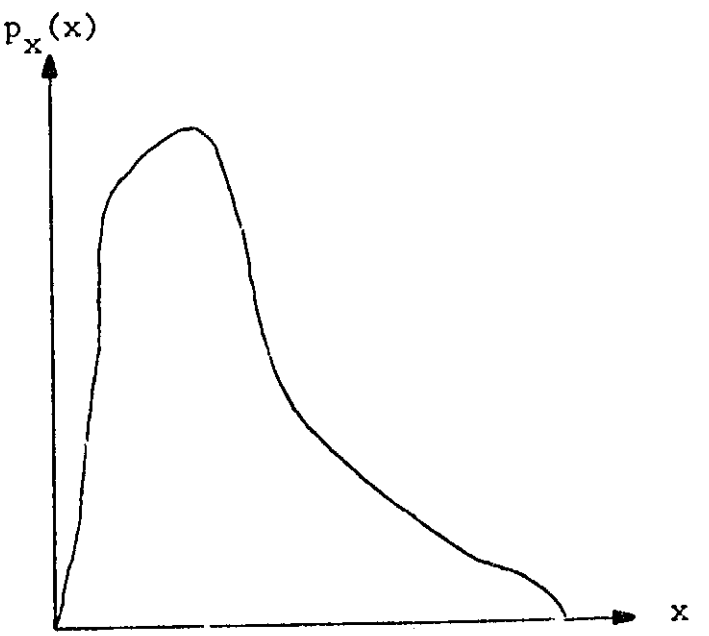
As will be seen below, the kurtosis of a histogram can be used to describe its peakedness about the mean. The coefficient of kurtosis that quantifies this peakedness is given by the relation

$$a_4 = \frac{m_4}{\sigma_x^4}. \tag{22}$$

For the normal density, this coefficient is three. For this reason, the coefficient of kurtosis is sometimes defined as



(a)



(b)

Figure 32. Examples of (a) negative skewness and (b) positive skewness.

$$a_4 = \frac{m_4}{\sigma_x^4} - 3. \qquad (23)$$

Using this relation, histograms can be separated into three categories. If the coefficient is negative, the histogram is termed platykurtic. An example of this appears in Figure 33(a). If the coefficient is zero, the histogram is mesokurtic and approximates the normal density. This corresponds to Figure 33(b). Figure 33(c) represents a leptokurtic histogram which has a positive coefficient of kurtosis.

The above development yields four parameters that can be used to describe a histogram. They give a method for describing the brightness and contrast of the image which are qualities that are visible. They also describe the asymmetry and peakedness of the histogram which, although not readily interpretable in terms of visual quality, relate to the physical description of the histogram. Using the four statistical parameters described above as a starting point, a method for specifying histograms interactively can be formulated. The basic approach is based on controlling the specified histogram by manipulations which are related to these parameters.

Four parameters are used in the histogram specification method described below. The first parameter used is denoted by m. This specifies a point in the pixel spectrum about which the histogram will be generated. The second parameter is denoted by h and specifies the height of the histogram at

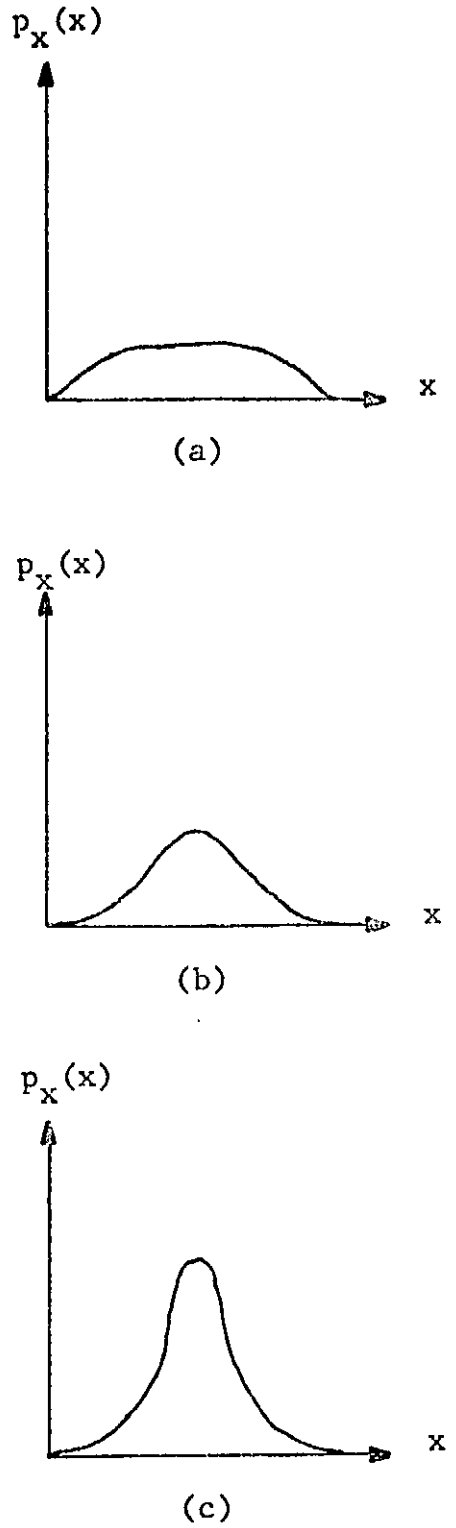


Figure 33. Examples of (a) platykurtic, (b) mesokurtic, and (c) leptokurtic densities.

the point m. The third and fourth parameters are referred to as the right spread (θ_R) and the left spread (θ_L) about m. It is noted that m is basically related to the mean of the histogram and h to its kurtosis. The other two parameters control spread (variance) and symmetry. Although the number of parameters could easily be increased, experiments have indicated that more than four parameters are quite difficult to control interactively, particularly when enhancement speed is an important consideration.

There are two basic cases that illustrate the manner in which the histograms are specified. Figures 34(a) and 34(b) illustrate the approach for decreasing and increasing the spread about m. Both of these cases involve specifying the point m which can be located anywhere on the interval [0,1]. Then the height is chosen at that point and may assume any value greater than or equal to zero.

For the case of decreasing spread, the parameters θ_R and θ_L specify the angle from vertical, as shown in Figure 34(a). These parameters can assume values from zero degrees to 90 degrees. Point j moves along the line segment that connects (0,1) and (m,0) as θ_L varies, while point k moves between (1,1) and (m,0) as θ_R varies. For the case of increasing spread, θ_L and θ_R specify the points along the vertical axis at which the points j and k are fixed. The lower bound for these points is zero and theoretically there is no upper bound.

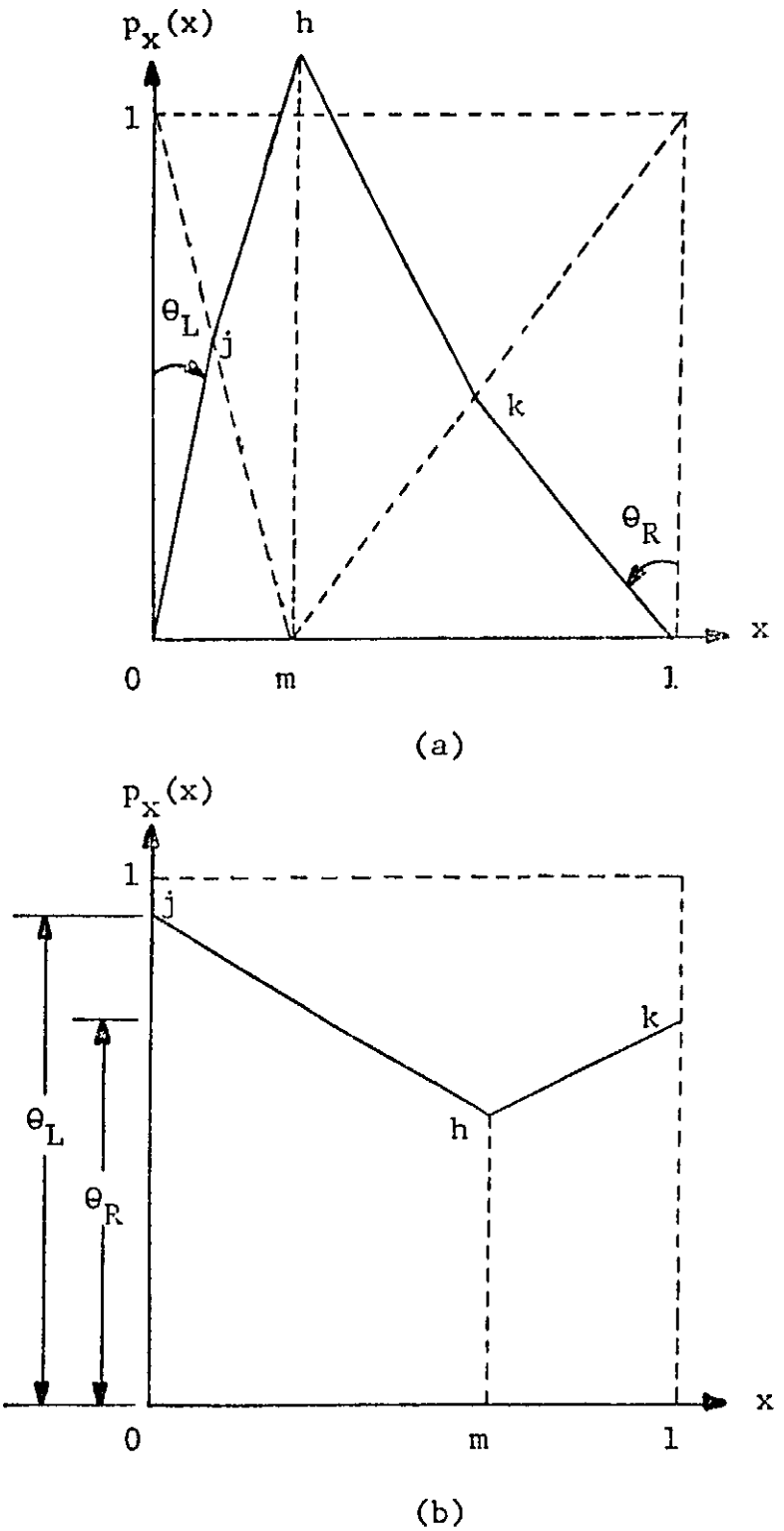
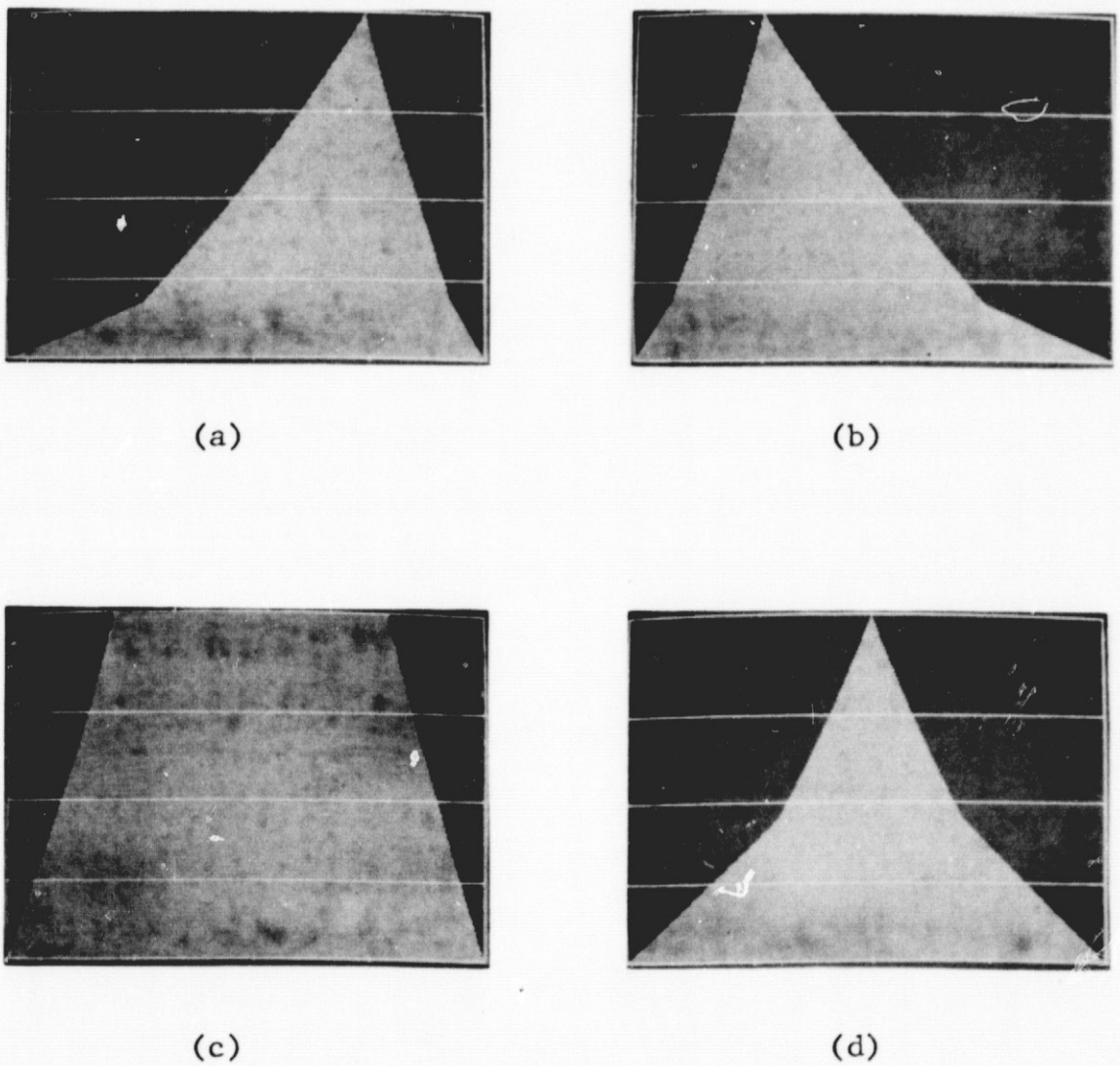


Figure 34. Methods for specifying histograms with (a) decreased and (b) increased spread.

The four parameters are specified such that the left and right sides of the histogram can be manipulated independently. Once the parameters have been specified, a piece-wise linear curve can be constructed. The curve is then normalized to unity area while preserving the shape of the histogram. Figure 35 illustrates four curves that were generated with this specification method. Figures 35(a) and 35(b) illustrate positive and negative skewness as in Figures 32(a) and 32(b), page 45. Figures 35(c) and 35(d) illustrate platykurtic and leptokurtic histograms as in Figures 33(a) and 33(b), page 47.

The above approach was implemented on a D.E.C. PDP 11/40 minicomputer, using four system potentiometers which are interfaced to the processor. Each potentiometer controls one of the four parameters described above by varying a voltage input to an analog to digital converter. This voltage is then sampled and expressed as a number between +1 and -1. If this algorithm were to be implemented in a real time image enhancement system, the potentiometers could be replaced by a joystick with four degrees of freedom as illustrated in Figure 36. This would be more convenient than the potentiometers, and an operator with little or no knowledge of enhancement methods or the structure of the algorithm can rapidly learn to manipulate the joystick in order to achieve fast enhancement results.



(a) (b)

(c) (d)

Figure 35. Example histograms generated with the method described in Chapter III.

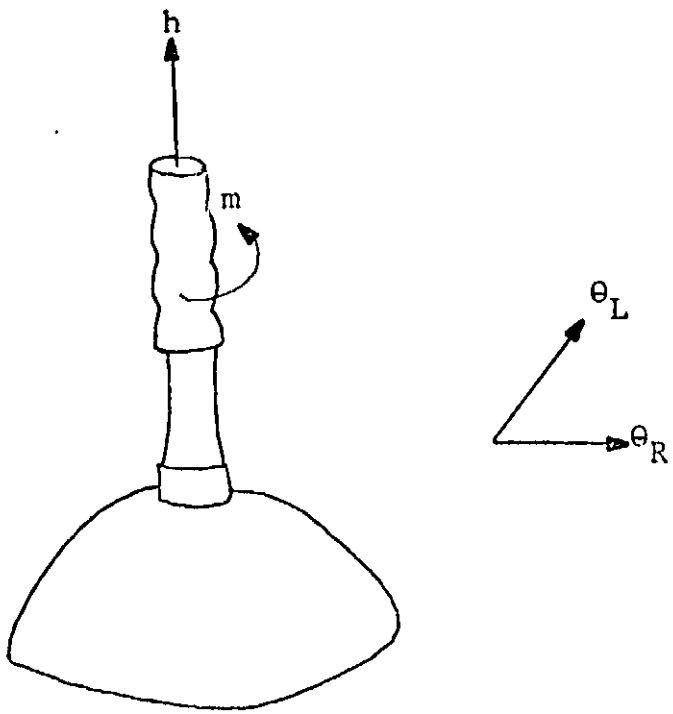


Figure 36. Joystick with four degrees of freedom for controlling the histogram specification process.

CHAPTER IV

EXPERIMENTAL RESULTS

This chapter presents several results of processing with the direct histogram specification algorithm developed in Chapters II and III. The pictures presented in the following pages, as well as those of previous chapters, are all 256 x 256 or medium-sized pictures. They are displayed in a 480 x 512 format, which constitutes a standard frame in a 525-line television monitor. The pictures are displayed with 256 gray levels with no scaling or translation of the data before display.

For each experiment, three pictures will be presented: (1) the original image, (2) the histogram-equalized image, and (3) the image that results from the histogram specification algorithm. Each of the pictures resulting from the specification algorithm was generated using two iterations of the preprocessing technique described in Chapter II. The histograms of the original image, the histogram-equalized image, the desired image and the resulting image are also presented. They will be referred to as parts (d), (e), (f) and (g), respectively.

The four parameters used to generate the desired histogram of each picture (see Chapter III) are listed in Table I.

TABLE I

TABLE OF HISTOGRAM PARAMETERS USED TO GENERATE THE RESULTS OF CHAPTER IV

Figure Number	m $0 \leq m \leq 1$	θ_L [a]	θ_R [a]	h $0 \leq h$
37	.350	10.67*	14.00	0.32
38	.245	4.91*	7.44*	7.02
39	.630	2.42*	6.98*	4.19
40	.770	80.10	9.21*	1.07
43	.705	3.89*	78.30	0.78
44	.100	49.50	38.70	0.40

[a] Values of θ_L and θ_R denoted by an asterisk are to be interpreted as displacements along the vertical axis. (See Figure 34, page 49.) Other values of θ_L and θ_R are angles from vertical in degrees.

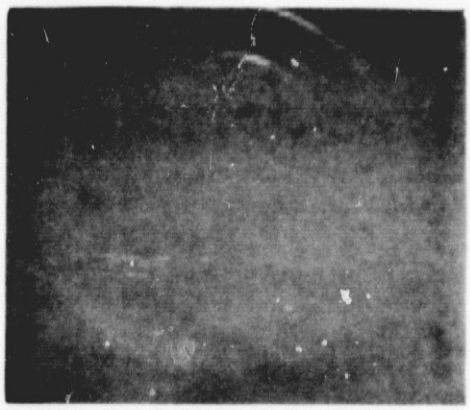
The values of θ_L and θ_R that are denoted with an asterisk are to be interpreted as displacements along the vertical axis. (See Figure 34, page 49.) The other values are to be interpreted as angles from vertical in degrees.

Result No. 1

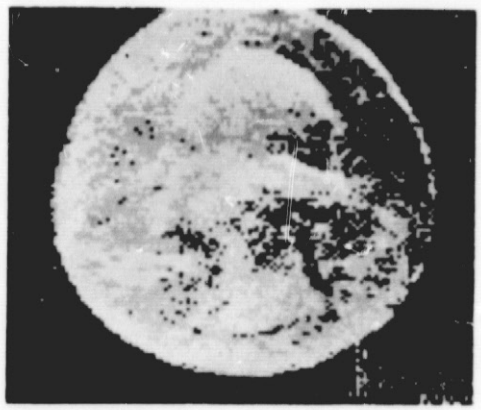
Figure 37(a) is the same quarter shown in Figure 11, page 24. It is repeated here to illustrate the use of the preprocessor. Figure 37(b) repeats the histogram equalization of the original picture. Figure 37(c) is the result of processing the preprocessed picture with the direct histogram specification method. Note that the word "LIBERTY" is discernible, whereas in the equalization case, it is almost completely obliterated. Figures 37(d) through 37(g) illustrate the histograms as outlined above.

Result No. 2

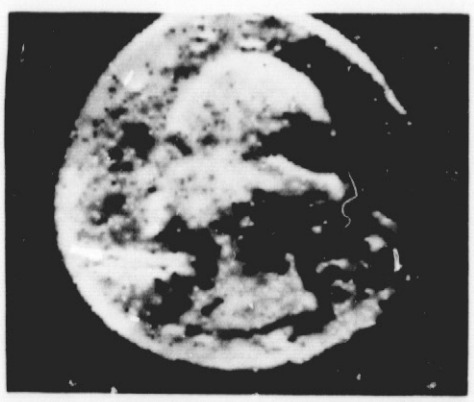
Figure 38(a) is the image of a dark room as viewed from the doorway. The striped shirt of a subject is just barely visible. Histogram equalization produces the image in Figure 38(b). Note that the contrast has been increased. Note also that the subject at the table has been enhanced, with the chairs at the table partly visible. Further processing with the specification algorithm yields the image in Figure 38(c). Note that the chairs have been enhanced further and the details of the objects on the table have also been brought out. The histograms for this example are illustrated in Figures 38(d) through 38(g).



(a)



(b)



(c)

(g)

(f)

(e)

(d)

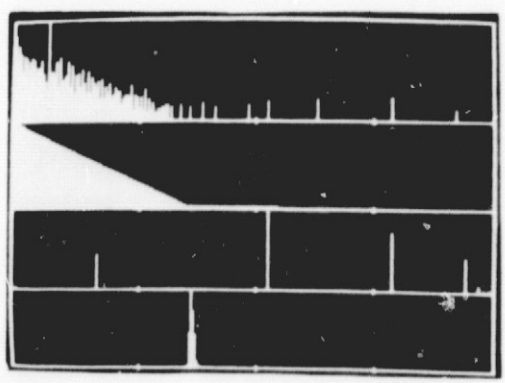
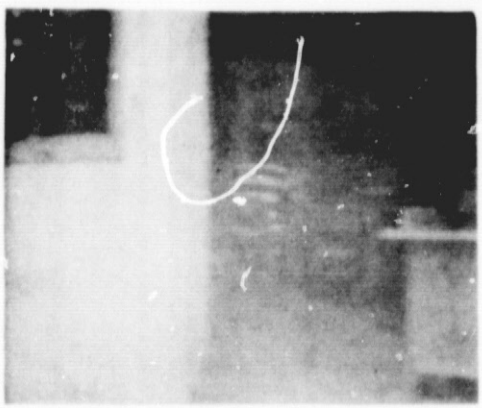
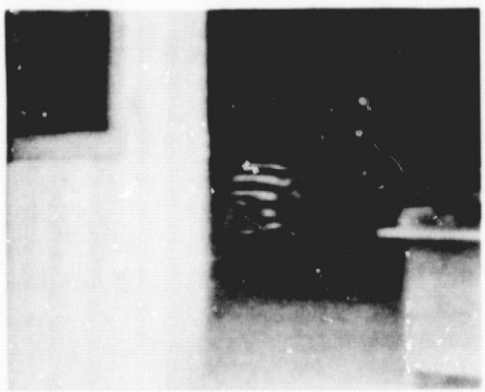


Figure 37. Results of processing the poorly illuminated image of a quarter.



(a)

(b)



(c)

(g)

(f)

(e)

(d)

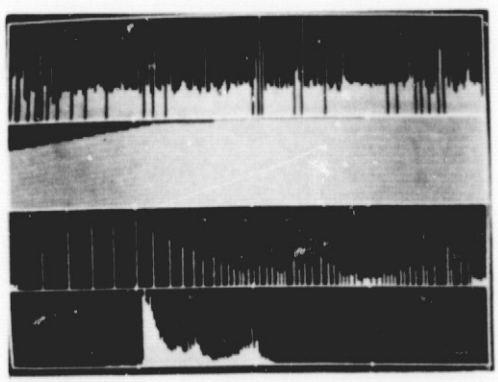


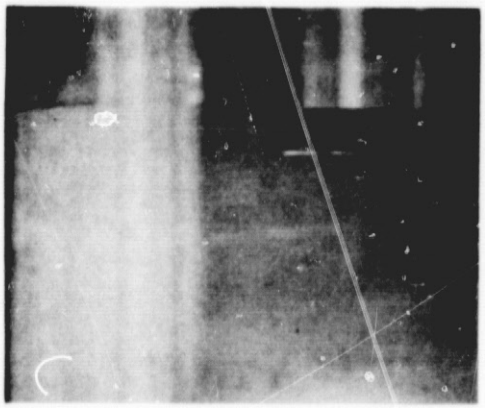
Figure 38. Results of processing the image of a laboratory with a student inside.

Result No. 3

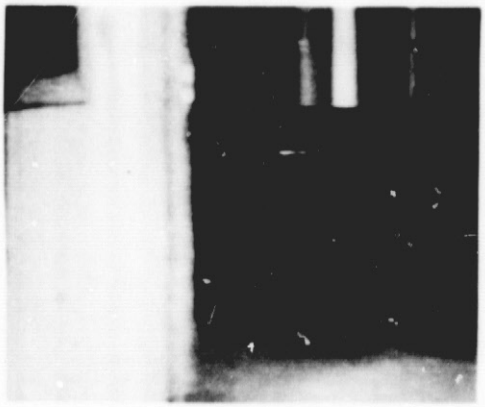
Figure 39 illustrates another view of the same room illustrated in Figure 38. In the original, Figure 39(a), there is not much contrast. Histogram equalization has increased the contrast, as illustrated in Figure 39(b), but has failed to bring out the detail in the room. Figure 39(c) illustrates the result of processing with the histogram specification algorithm. Note that the chairs in the room and some objects on the tables have been brought out in this example. Figures 39(d) through 39(g) illustrate the histograms pertinent to this example.

Result No. 4

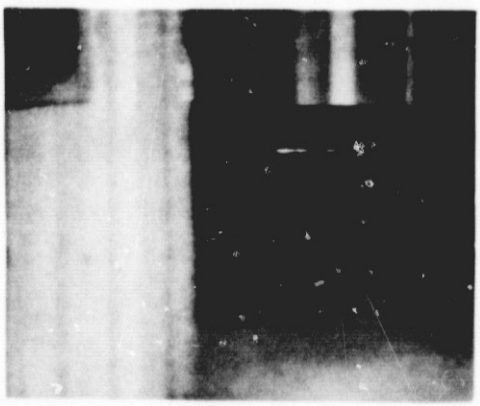
Figure 40(a) is a picture of a truck taken against a very light background. The brightness of the background has forced the dark details of the truck to have relatively very low contrast. Histogram equalization, in this case, served only to increase the contrast between the truck and the background, as illustrated in Figure 40(b). Processing with the histogram specification algorithm produced the image in Figure 40(c). The reason for the improvement in this picture is that the specification algorithm allowed the user to confine the background pixels to a very small portion of the spectrum, while spreading the dark pixels of the truck over the majority of the spectrum. Histogram equalization, on the other hand, spread the background pixels over most of the spectrum, leaving the important information in a very



(a)



(b)



(c)

(g)

(f)

(e)

(d)

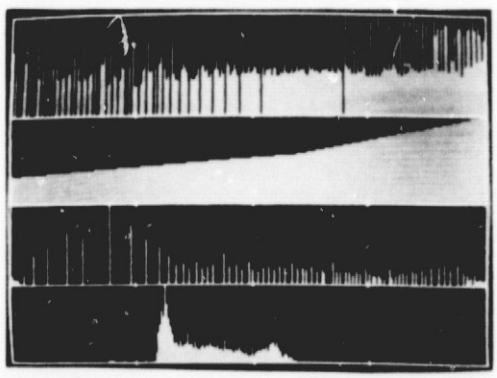
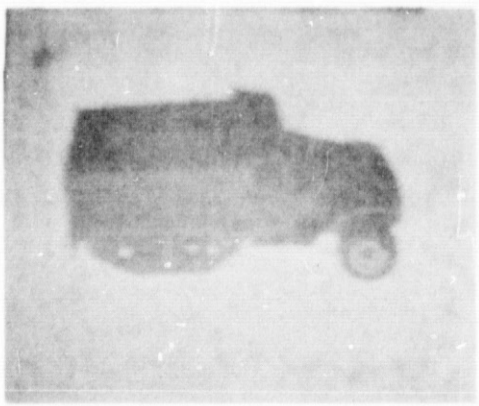
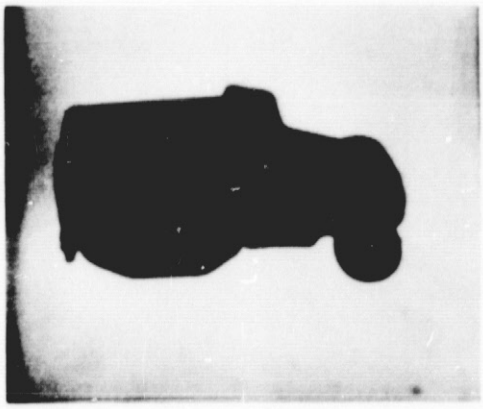


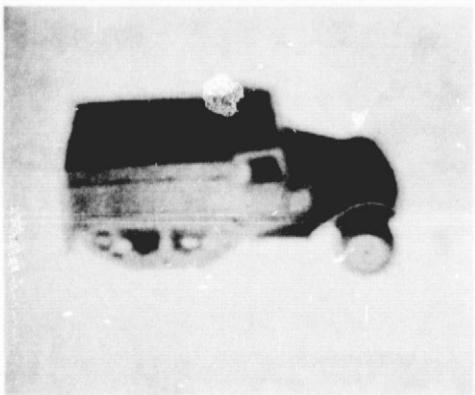
Figure 39. Results of processing the image of a laboratory taken from the door.



(a)



(b)



(c)

(g)

(f)

(e)

(d)

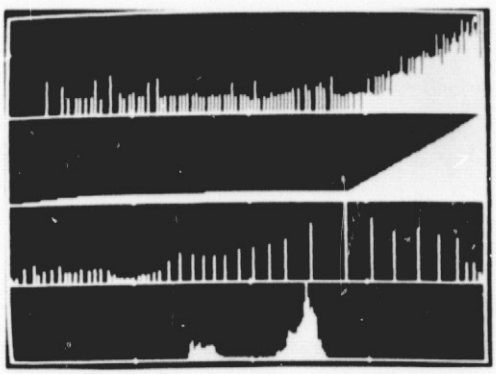


Figure 40. Results of processing the image of a truck taken against a light background.

small region. This is illustrated in the histograms in Figures 40(d) through 40(g).

It was mentioned in Chapter III that one way of specifying a histogram is to digitize a curve that has been sketched. The curve in Figure 41 was digitized using a television camera and a joystick. The program for doing this is listed in the appendix. Figure 42(a) illustrates the result of specifying the histogram in Figure 41. Figures 42(b) through 42(e) illustrate the relevant histograms in the same order as described above.

Result No. 5

The above examples illustrate images whose detail was enhanced by the use of the histogram specification algorithm. There are some cases, however, where histogram equalization produces an acceptable image. Figure 43 illustrates one of these examples. Figure 43(a) illustrates the image of the same truck as in Figure 40(a). This image, however, was taken against a dark background with very poor lighting. Histogram equalization succeeded in bringing out the details of the truck, as illustrated in Figure 43(b). Processing with the direct histogram specification method did not improve the image significantly, as can be seen in Figure 43(c). It should be kept in mind that this approach will never do any worse than histogram equalization, since histogram equalization is a special case of the direct histogram specification method. The pertinent histograms are in Figures 43(d) through 43(g).

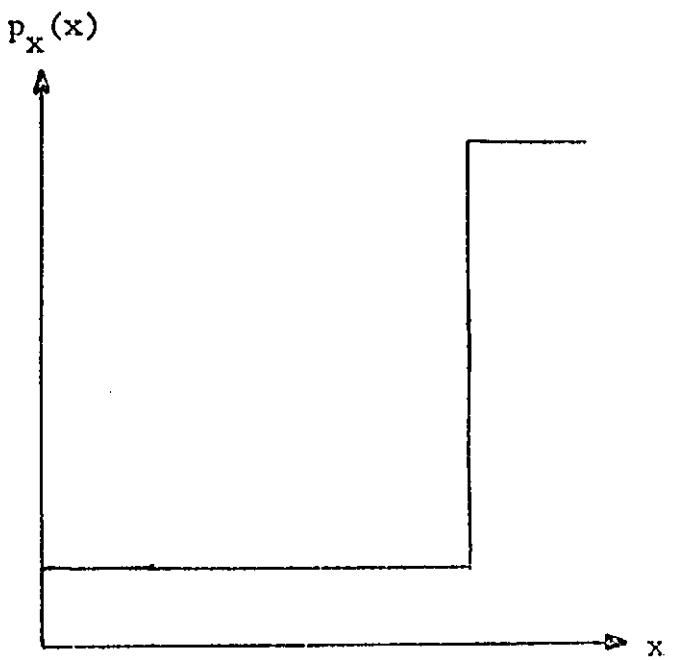
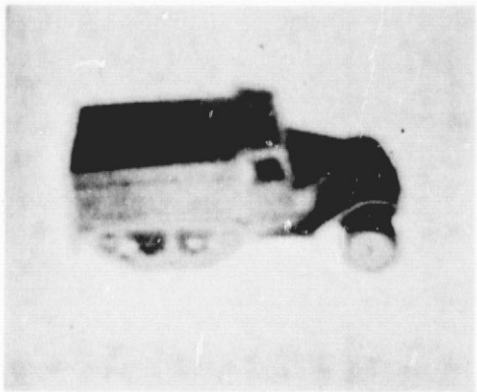


Figure 41. Histogram that was input to the computer with a joystick-cursor arrangement.



(a)

(e)

(d)

(c)

(b)

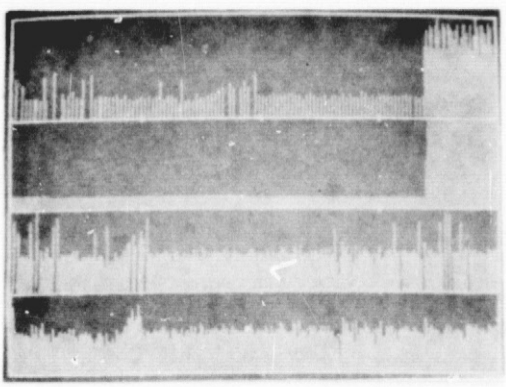
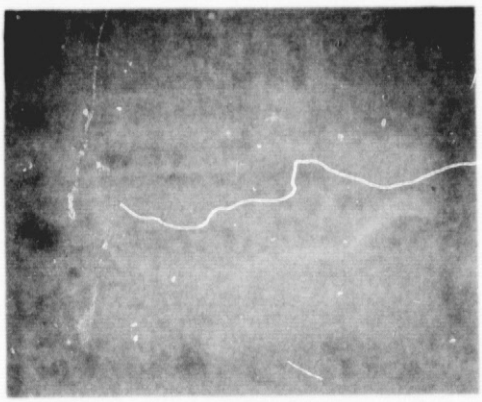
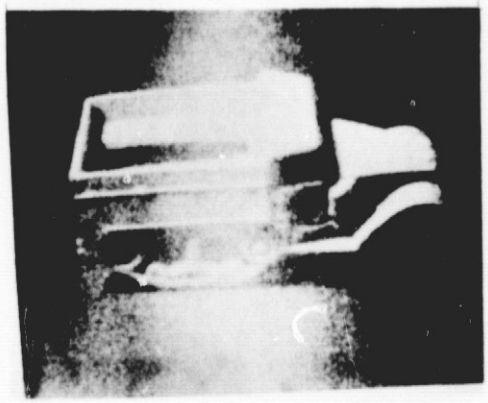


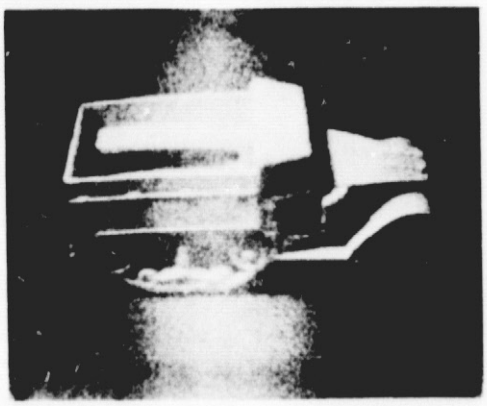
Figure 42. Results of processing the truck in Figure 40(a) with the histogram shown in Figure 41.



(a)



(b)



(c)

(g)

(f)

(e)

(d)

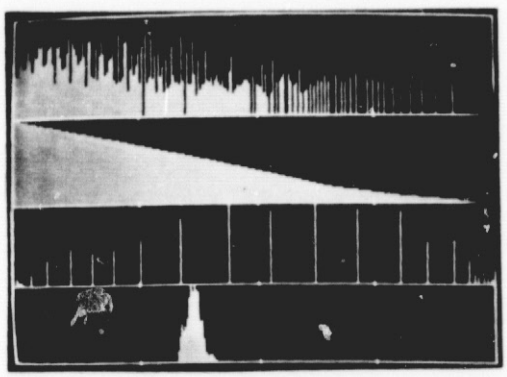
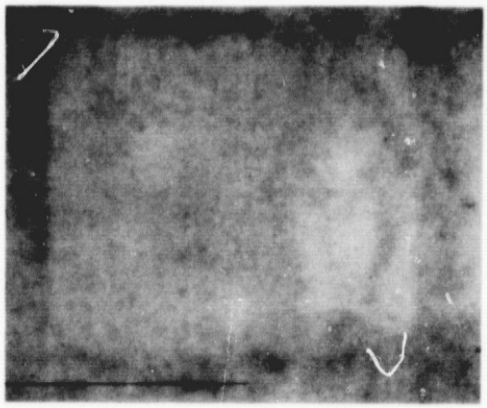


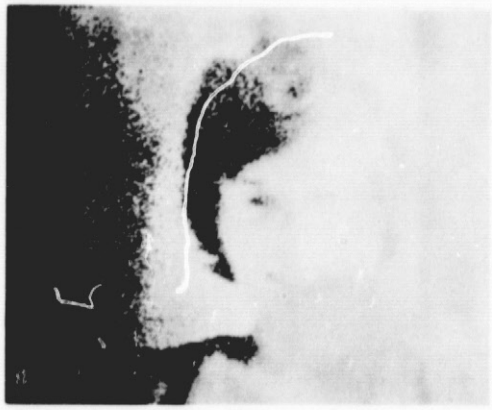
Figure 43. Result of processing a low contrast picture of a truck.

Result No. 6

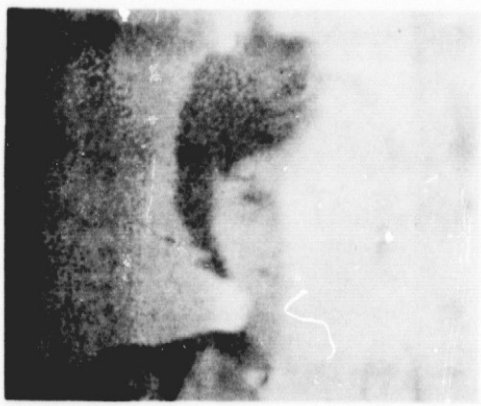
As a final result, consider Figure 44. Figure 44(a) is the image of a woman's face covered with a shadow. This picture was taken from a photograph placed in a shadow. Note that the histogram equalization of this image produces a very noisy result, as seen in Figure 44(b). The direct histogram specification method was effective in decreasing the prominence of the noise, as is illustrated in Figure 44(c). The relevant histograms for this example are in Figures 44(d) through 44(g).



(a)



(b)



(c)

(g)

(f)

(e)

(d)

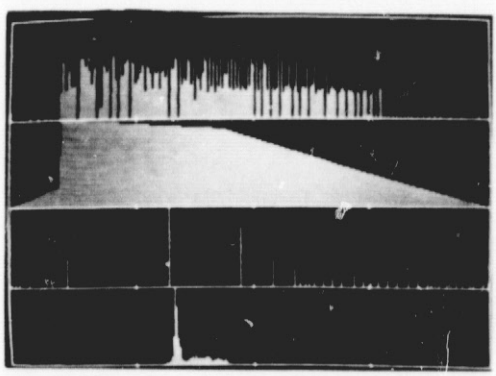


Figure 44. Result of processing the image of a woman's face.

CHAPTER V

CONCLUSIONS

The direct histogram specification method developed in Chapters II and III has been shown to be a valuable tool for interactive image enhancement. As was indicated in Chapter IV, this method can often improve the results obtainable with the well-known histogram equalization approach. Some experiments were described where histogram equalization yielded enhanced images that were quite acceptable. Since this method is a special case of the algorithm developed in this paper, however, the direct specification method will always do at least as well as the classical equalization approach.

The technique described in Chapter III for interactively specifying a histogram is just one of many that could have been implemented. Although the results obtained with this approach have proved satisfactory, other methods should be investigated and compared in terms of their interactive features. If this algorithm is to be implemented in a real-time system, the specification method needs also to be investigated with respect to its computational complexity. Its present form is suitable for the type of processing done here, but it may need to be simplified somewhat, in order to decrease the computation time required.

The histogram smoothing preprocessor presented in Chapter II has proven useful in reducing the error between the specified and resulting image histograms. Experiments have indicated, however, that the preprocessed images that result from the histogram specification enhancement do not differ radically from those that were not preprocessed. The preprocessor does not appear to have much impact on the final visual results. Theoretically, however, it is of great significance if a uniform histogram is desired. Further work in this area could include the possibility of weighting the points in the neighborhood when performing the summation in Equation (16), page 30.[13]

Ultimately, this algorithm could be implemented in a real-time or almost real-time interactive image enhancement system. There are four principal areas that must be considered:

(1) calculation of the image histogram,

(2) computation of the transformation function,

(3) computation of the new pixel values, and

(4) reconstruction of the image for display.

There are two approaches that can be taken. One is to use a digital system, the other a hybrid system.

To process the image digitally (like all of the experiments presented in this paper), the first three areas mentioned above can be implemented with a special purpose computer. The fourth area involves the specification of a

system that will convert the digital data to an analog video signal for display on a monitor. The major problem with this approach would probably be the time required to retrieve the original image from storage and output the new image to the display device. This has been the limiting factor in the work reported in this paper when attempting to simulate a real-time system.

The most attractive approach is to process the video signal in a hybrid system. This alleviates the problem of reconstructing the image, as well as the computation of the new pixel values. The idea would be to generate an analog transfer function that is digitally controlled, as illustrated in Figure 45. The transfer function could be made up of a network of diode function generators that generate a piece-wise linear approximation to the transformation function.(14)

A diode function generator has an input-output voltage characteristic as illustrated in Figure 46. The breakpoint, Eb, is variable as well as the slope, S. In a transfer function generator, the breakpoints could be fixed based on the number of specified segments. Then the slope of each segment could be controlled digitally, using either a servo-controlled potentiometer or a solid state digital coefficient unit.

The digital control could easily be supplied by an off-the-shelf microprocessor. Its function would be to

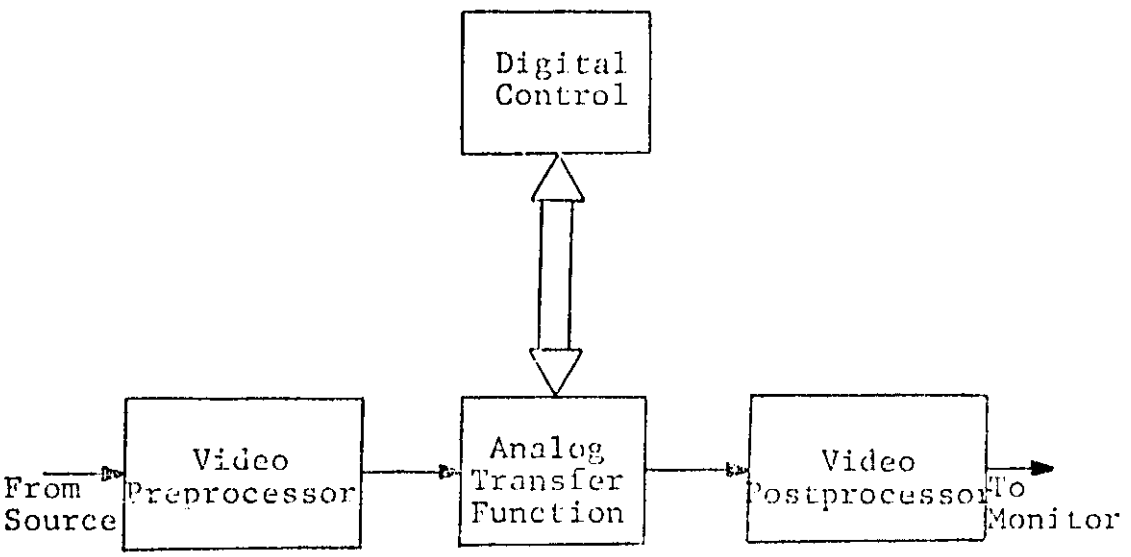


Figure 45. Block diagram of a hybrid implementation of an interactive image enhancement system.

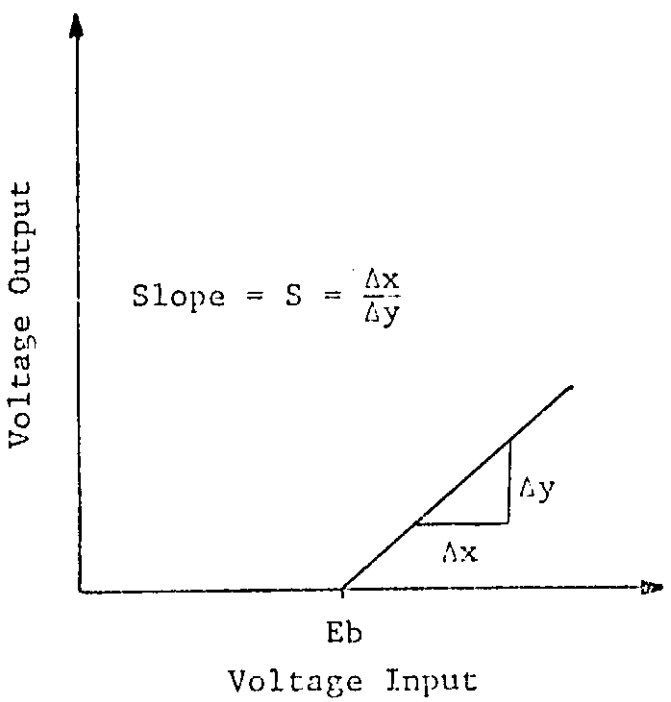


Figure 46. Input-output characteristics of a diode function generator.

sample a part of the video signal to compute a histogram representative of the image at that point in time. Using this information, a transformation function can then be computed and the slope coefficients in the transfer function can be set.

The hybrid approach has several advantages. First, it does not require a complex display system to generate a video signal, since it operates on the video directly. Second, since the picture does not need to be stored, the read/write memory requirements for the system are minimal. The algorithm can be stored in read-only memory. The only random access memory required would be to store the histogram and the computed values of the transfer function. This requirement could be simplified considerably by decreasing the spatial resolution and number of gray levels when obtaining the image histogram.

In summary, an image enhancement algorithm has been developed and implemented. This interactive algorithm, which is based on directly specifying a histogram, has been shown to yield results that are often superior to those obtainable by the popular histogram equalization technique. An algorithm has also been implemented for improving the uniformity of the histogram. Finally, two methods have been proposed for implementing the direct histogram specification method of enhancement in a real-time interactive image enhancement system.

BIBLIOGRAPHY

BIBLIOGRAPHY

1. Haralick, R. M., Shanmugan, K. and Dinstein, I., "Textural Features for Image Classification," IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-3, November 1973, p. 610.

2. Huang, T. S., Schrieber, W. F. and Tretiak, O. J., "Image Processing," Proceedings of the IEEE, Vol. 59, November 1971, pp. 1586-1609.

3. Rosenfeld, A. and Troy, E. B., "Visual Texture Analysis," Conference Record of the Symposium on Feature Extraction and Selection in Pattern Recognition, IEEE Publication 70C 51-C, Argonne, Illinois, October 1970.

4. Stockham, T. G., "Image Processing in the Context of a Visual Model," Proceedings of the IEEE, Vol. 60, July 1972, p. 828.

LIST OF REFERENCES

LIST OF REFERENCES

1. Thompson, D. D., "Image Processing System Hardware Description," Technical Report TR-EE/CS-75-15, Electrical Engineering Department, University of Tennessee, Knoxville, Tennessee, 1975.

2. Thompson, D. D., "Image Processing System Software Description," Technical Report TR-EE/CS-75-16, Electrical Engineering Department, University of Tennessee, Knoxville, Tennessee, 1975.

3. Papoulis, A., Probability, Random Variables, and Stochastic Processes, McGraw-Hill Book Co., New York, New York, 1965.

4. Andrews, H. C., Tescher, A. G. and Krueger, R. P., "Image Processing by Digital Computer," IEEE Spectrum, Vol. 9, No. 7, July 1972, pp. 20-32.

5. Hall, E. L., et. al., "A Survey of Preprocessing and Feature Extraction Techniques for Radiographic Images," IEEE Transactions on Computers, Vol. C-20, No. 9, September 1971, pp. 1032-1044.

6. Hall, E. L., "Almost Uniform Distributions for Computer Image Enhancement," IEEE Transactions on Computers, Vol. C-23, No. 2, February 1974, pp. 207-208.

7. Andrews, H. C., "Digital Image Restoration: A Survey," Computer, Vol. 7, No. 5, May 1974, pp. 36-45.

8. Gonzalez, R. C. and Birdwell, J. D., "The Application of Image Enhancement Techniques to Remote Manipulator Operation," Technical Report TR-EE/CS-74-15, Electrical Engineering Department University of Tennessee, Knoxville, Tennessee, 1974.

9. Gonzalez, R. C. and Fittes, B. A., "Gray-Level Transformations for Interactive Image Enhancement," Proceedings of the Conference on Remotely Manned Systems, June 1975, pp. 17-19.

10. Troy, E. B., Deutsch, E. S. and Rosenfeld, A., "Gray-Level Manipulation Experiments for Texture Analysis," IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-3, No. 1, January 1973, pp. 91-98.

11. Spiegel, M. R., Statistics, Schaum Publishing Co., New York, New York, 1961.

12. Selby, S. M., Editor, Standard Mathematical Tables, The Chemical Rubber Co., Cleveland, Ohio, 1971.

13. Hummel, R. A., "Histogram Modification Techniques," Technical Report TR-329, University of Maryland, College Park, Maryland, 1974.

14. Korn, G. A. and Korn, T. M., Electronic Analog and Hybrid Computers, McGraw-Hill Book Co., New York, New York, 1972.

APPENDIX

APPENDIX

INTERACTIVE IMAGE ENHANCEMENT SYSTEM PROGRAM

This appendix describes the interactive image enhancement system program implemented in conjunction with this paper. Figure 47 illustrates a simplified flow chart of the program logic.

This program was written to run under the RT-11 Operating System on the D.E.C. PDP 11/40. The main program and most of the subroutines were written in RT-11 FORTRAN IV. The other subroutines were written in PDP 11/40 assembler language.

Three system subroutines were used. 'ASSIGN' is used in conjunction with the DEFINE FILE statement to open a certain direct access file. 'DATE' returns the current date in as ASCII character string. 'IDATE' returns the current date in three integers denoting month, day and year.

The program listing follows Figure 47. The main program is listed first followed by the FORTRAN subroutines in alphabetical order. Finally, the two assembler language subroutines are listed.

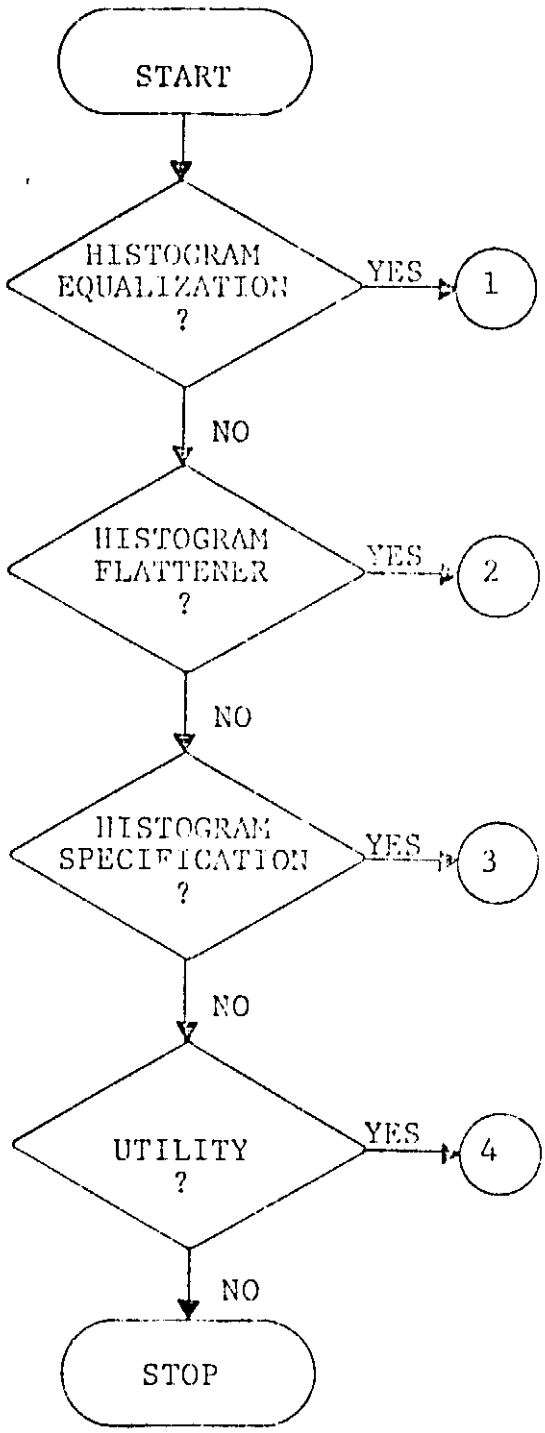


Figure 47. Simplified flow chart of the interactive image enhancement system program.

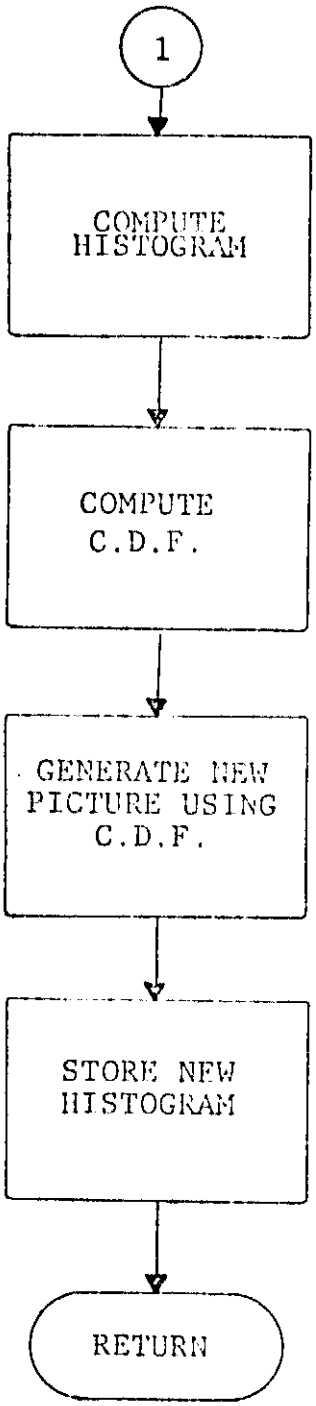


Figure 47. (continued)

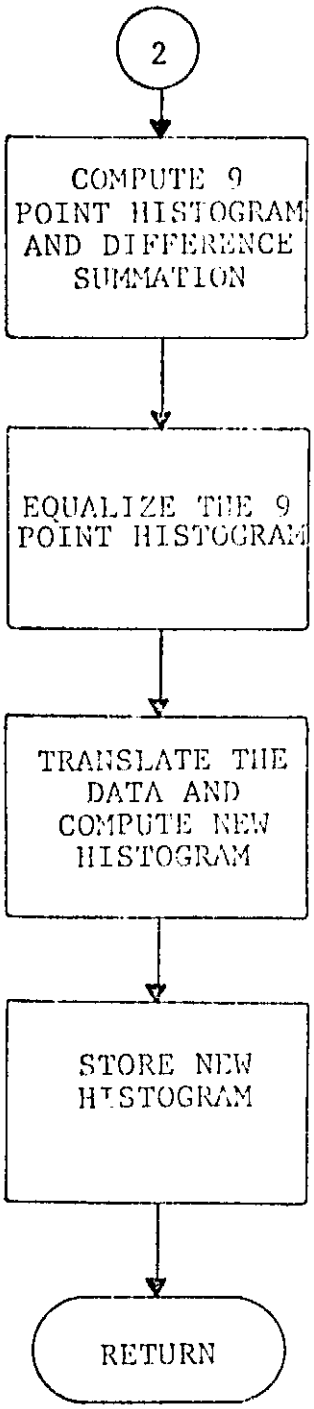


Figure 47. (continued)

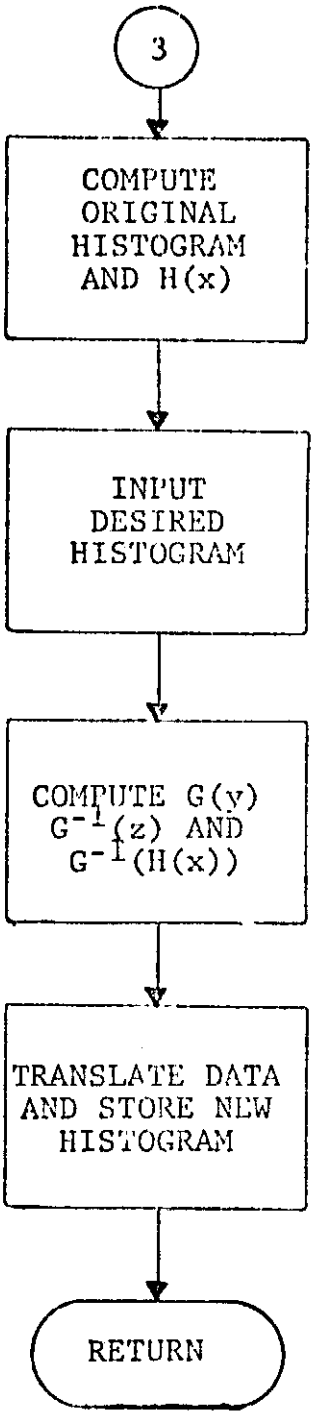


Figure 47. (continued)

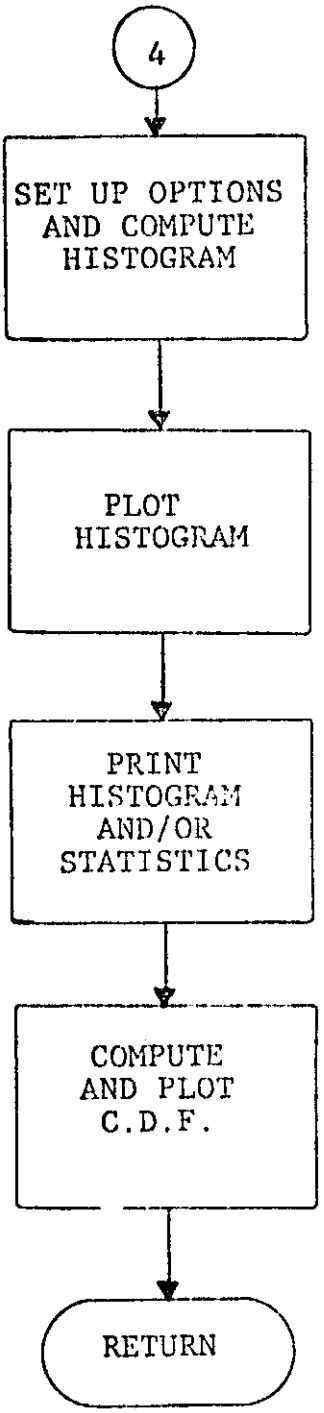


Figure 47. (continued)

```
C          THIS PROGRAM IS INTENDED TO PERFORM AS AN
C          INTERACTIVE IMAGE ENHANCEMENT SYSTEM (IIES),
C          CAPABLE OF PROVIDING THE FOLLOWING SERVICES:
C               1)  HISTOGRAM EQUALIZATION,
C               2)  HISTOGRAM FLATTENING,
C               3)  HISTOGRAM SPECIFICATION ENHANCEMENT,
C               4)  UTILITIES.
C          IT PERFORMS IN A VERY SIMPLE QUESTION AND ANSWER
C          FORMAT WITH THE USER CAPABLE OF SPECIFYING THE
C          PRINTING OF HISTOGRAMS AND/OR STATISTICS, PLOTTING
C          OF HISTOGRAMS, PLOTTING TRANSFORMATIONS, ETC.
C          OTHER ALGORITHMS CAN BE EASILY ADDED TO IT SINCE
C          THE MAIN PROGRAM CONSISTS OF MOSTLY SUBROUTINE
C          CALLS THAT PERFORM THE VARIOUS BASIC FUNCTIONS.
C          EACH PROGRAM SECTION CONSISTS OF A SECTION OF
C          OPERATOR DIALOG THAT ALLOWS THE USER TO SELECT
C          CERTAIN OPTIONS, AS WELL AS THE CODE THAT ACTUALLY
C          PERFORMS THE FUNCTION.  THIS PROGRAM IS SET UP TO
C          INTERCEPT ALL DIRECT ACCESS I/O ERRORS.  IT PRINTS
C          A MESSAGE ON THE CONSOLE DEVICE AND MAINTAINS
C          CONTROL WITHIN THE PROGRAM.
C
C          SUBROUTINES PLOTS, NEWPEN AND VIDPLT ARE THOSE THAT
C          PERFORM PLOTTING ON THE VIDEO DISK.  THEY ARE
C          CALLED IN THE FOLLOWING MANNER:
C               CALL PLOTS (IBK); WHERE IBK IS ON THE INTERVAL
C               (0,15), 0 REPRESENTING BLACK AND 15
C               REPRESENTING WHITE.  THIS SUBROUTINE ERASES
C               THE SCREEN TO THE COLOR SPECIFIED BY IBK, AND
C               INITIALIZES THE PEN TO (0,0).  THE PEN COLOR
C               IS SET TO WHITE.
C               CALL NEWPEN (IPEN), WHERE IPEN IS ON THE
C               RANGE (0,15).  THIS CHANGES THE PEN COLOR.
C               CALL VIDPLT (IX,IY,IP); WHERE (IX,IY) ARE THE
C               X-Y COORDINATES WHERE THE PEN IS TO BE MOVED
C               TO.  IF IP IS +2, THE PEN WILL BE DOWN.  IF IT
C               IS +3, IT WILL BE UP.  IF IP IS -2, THE PEN
C               WILL BE DOWN AND A NEW ORIGIN DEFINED AT THAT
C               POINT.  IF IP IS -3, THE PEN WILL BE UP AND
C               A NEW ORIGIN DEFINED.
C
C          SUBROUTINES USED - CAMERA,CRVPLT,GM1HX,GRAYEQ,HIST,
C                             HSTCRV,HSTPLT,HSTPRT,HSTSTF,
C                             INVCRV,POTENT,RECNUM,RWERR,
C                             RWLBLO,STAT,TRHIST,UNIRMS,
C                             WRITEO,XLATE
C
C
```

```
C          THIS SECTION DEFINES THE ARRAYS THAT ARE USED
C          THROUGHOUT THE PROGRAM.  TO CONSERVE MEMORY, AS
C          MANY ARRAYS AS ARE NEEDED ARE PASSED TO THE
C          SUBROUTINES AS ARGUMENTS, EVEN IF THEY ARE NOT USED
C          AS INPUTS TO OR OUTPUTS FROM THE SUBROUTINE.  THIS
C          SECTION ALSO PERFORMS THE DIALOG NECESSARY TO
C          DETERMINE WHICH ROUTINE IS DESIRED, AND TRANSFERS
C          CONTROL TO THE APPROPRIATE PLACE.
C
C
      INTEGER*2 SYSDAT(1),IPIC(2048),PLTNUM,HSTPAS,CDFPAS
      INTEGER*2 ERFLG,XTAB(256),YTAB(256),ZTAB(256)
      REAL*4 XHIST(256),XT(11),YHIST(256),YT(11),ZHIST(256)
      REAL*4 ZT(11),THIST(256),TT(11),ST(4),GRHIST(2304)
      LOGICAL*1 YES,NO,DAT(9),ANS,INFIL(14),OUTFIL(14)
      LOGICAL*1 LPIC(4096),NAM1(8)
      LOGICAL TRUE,FALSE,PRS,PRH,PLH,PLC,TLD,GNC,POTS,OUTPUT
      LOGICAL OLDPRH
      DATA YES,NO,TRUE,FALSE/'Y','N',.TRUE.,.FALSE./
      EQUIVALENCE (IPIC(1),LPIC(1)),(GRHIST(1),IPIC(1537)),
     1(GRHIST(1025),YHIST(1)),(GRHIST(1281),ZHIST(1)),
     2(GRHIST(1537),THIST(1)),(GRHIST(1793),XTAB(1)),
     3(GRHIST(1921),YTAB(1))
      CALL DATE (DAT)
      CALL IDATE (I,J,K)
      SYSDAT(1)=(K-72)+32*(J+32*I)
      OLDPRH=FALSE
      WRITE (7,1)
  100 WRITE (7,2)
      READ (5,3) ANS
      IF (ANS.EQ.YES) GO TO 400
      WRITE (7,4)
      READ (5,3) ANS
      IF (ANS.EQ.YES) GO TO 700
      WRITE (7,5)
      READ (5,3) ANS
      IF (ANS.EQ.YES) GO TO 1000
      WRITE (7,6)
      READ (5,3) ANS
      IF (ANS.EQ.YES) GO TO 200
      STOP 'IIES FINISHED'
C
C
C          THIS SECTION OF CODE IS THE UTILITY PROCESSOR.  IT
C          CAN PLOT AND PRINT HISTOGRAMS, PRINT STATISTICS
C          AND PLOT CUMULATIVE DISTRIBUTION FUNCTIONS.
C
C
  200 WRITE (7,7)
      HSTPAS=0
      PLTNUM=0
  210 WRITE (7,8)
```

```
      READ (5,3) INFIL
      IF (INFIL(1).EQ.32) GO TO 9999
      CALL ASSIGN (1,INFIL,14,'OLD','NC',1)
      DEFINE FILE 1 (130,1024,U,I1)
      CALL RECNUM (LPIC,NREC,NAM1,1,ERFLG)
      IF ((NREC.EQ.0).OR.(ERFLG.NE.0)) GO TO 300
      WRITE (7,9)
      READ (5,3) ANS
      PLH=FALSE
      IF (ANS.EQ.YES) PLH=TRUE
      IF (.NOT.PLH) GO TO 220
      WRITE (7,10)
      READ (5,3) ANS
      IF (ANS.NE.YES) GO TO 220
      WRITE (7,11)
      READ (5,12) PLTNUM
      PLTNUM=MIN0(PLTNUM,4)
      PLTNUM=MAX0(PLTNUM,1)
      HSTPAS=0
  220 HSTPAS=HSTPAS+1
      WRITE (7,13)
      READ(5,3) ANS
      PRH=FALSE
      IF (ANS.EQ.YES) PRH=TRUE
      IF ((.NOT.OLDPRH).AND.PRH) WRITE (6,16)
      WRITE (7,14)
      READ (5,3) ANS
      PRS=FALSE
      IF (ANS.EQ.YES) PRS=TRUE
      WRITE (7,21)
      READ (5,3) ANS
      PLC=FALSE
      IF (ANS.EQ.YES) PLC=TRUE
      IF (.NOT.PLC) GO TO 230
      WRITE (7,10)
      READ (5,3) ANS
      CDFPAS=0
      IF (ANS.EQ.YES) CDFPAS=1
  230 CALL HIST (NREC,1,ERFLG,IPIC,XHIST)
      IF (ERFLG.NE.0) GO TO 300
      CALL STAT (XT,XHIST)
      CALL UNIRMS (NREC,XHIST,ERR)
  240 IF (.NOT.(PRS.OR.PRH)) GO TO 260
  250 WRITE (6,17) NAM1,DAT
  260 IF (.NOT.PLH) GO TO 270
      CALL HSTPLT (PLTNUM,HSTPAS,XHIST)
  270 IF (.NOT.PRS) GO TO 280
      WRITE (6,18) XT
      WRITE (6,25) ERR
      WRITE (6,15)
  280 IF (.NOT.PRH) GO TO 290
      CALL HSTPRT (XHIST)
```

```
      WRITE (6,16)
  290 IF (.NOT.PLC) GO TO 300
      CALL HSTCRV (NREC,XTAB,XHIST)
      IF (PLH) PAUSE 'TYPE C/R FOR PLOT OF C.D.F.'
      CALL CRVPLT (CDFPAS,2,XTAB)
  300 ENDFILE 1
      OLDPRH=PRH
      WRITE (7,19)
      READ (5,3) ANS
      IF (ANS.EQ.YES) GO TO 100
      GO TO 210
C
C
C        THIS SECTION IS THE HISTOGRAM EQUALIZATION
C        PROCESSOR.  THE OPTIONS AVAILABLE ARE PLOTTING AND
C        PRINTING OF HISTOGRAMS, PRINTING OF STATISTICS AND
C        PLOTTING OF THE CUMULATIVE DISTRIBUTION FUNCTION.
C        FIRST THE LABEL RECORD IS COPIED, THEN THE
C        HISTOGRAM COMPUTED.  THEN THE CUMULATIVE
C        DISTRIBUTION FUNCTION IS CALCULATED AND THE PICTURE
C        IS TRANSLATED.  MEANWHILE, THE OPTIONS ARE
C        PERFORMED AS REQUESTED.  THEN THE NEW HISTOGRAM IS
C        STORED IN THE LABEL RECORD.
C
C
  400 WRITE (7,20)
  410 WRITE (7,9)
      READ (5,3) ANS
      PLH=FALSE
      IF (ANS.EQ.YES) PLH=TRUE
      WRITE (7,13)
      READ (5,3) ANS
      PRH=FALSE
      IF (ANS.EQ.YES) PRH=TRUE
      IF ((.NOT.OLDPRH).AND.PRH) WRITE (6,16)
      WRITE (7,14)
      READ (5,3) ANS
      PRS=FALSE
      IF (ANS.EQ.YES) PRS=TRUE
      WRITE (7,21)
      READ (5,3) ANS
      PLC=FALSE
      IF (ANS.EQ.YES) PLC=TRUE
  420 WRITE (7,8)
      READ (5,3) INFIL
      IF (INFIL(1).EQ.32) GO TO 9999
      CALL ASSIGN (1,INFIL,14,'OLD','NC',1)
      DEFINE FILE 1 (130,1024,U,I1)
      CALL RECNUM (LPIC,NREC,NAM1,1,ERFLG)
      IF ((NREC.EQ.0).OR.(ERFLG.NE.0)) GO TO 440
      WRITE (7,22)
      READ (5,3) OUTFIL
```

2

```
      IF (OUTFIL(1).EQ.32) GO TO 440
      CALL ASSIGN (2,OUTFIL,14,'NEW','NC',1)
      DEFINE FILE 2 (NREC,1024,U,I2)
      CALL RWLBLO (1,2,'$',SYSDAT,LPIC,ERFLG)
      IF (ERFLG.NE.0) GO TO 430
      CALL HIST (NREC,1,ERFLG,IPIC,XHIST)
      IF (ERFLG.EQ.0) GO TO 450
430   ENDFILE 2
440   ENDFILE 1
      GO TO 550
450   WRITE (7,23)
      CALL STAT (XT,XHIST)
      CALL HSTCRV (NREC,XTAB,XHIST)
      CALL TRHIST (XTAB,XHIST,YHIST)
      WRITE (7,24)
      CALL STAT (YT,YHIST)
      CALL UNIRMS (NREC,YHIST,ERR)
      IF (.NOT.PLH) GO TO 460
      CALL HSTPLT (2,1,XHIST)
      CALL HSTPLT (2,2,YHIST)
460   CALL XLATE (NREC,1,2,ERFLG,IPIC,XTAB)
      IF (ERFLG.NE.0) GO TO 430
      CALL HSTSTF (2,YHIST,ERFLG,IPIC)
      IF (ERFLG.NE.0) GO TO 430
      ENDFILE 1
      ENDFILE 2
470   IF (.NOT.(PRS.OR.PRH)) GO TO 540
480   WRITE (6,17) NAM1,DAT
      WRITE (6,23)
490   IF (.NOT.PRS) GO TO 500
      WRITE (6,18) XT
      WRITE (6,15)
500   IF (.NOT.PRH) GO TO 510
      CALL HSTPRT (XHIST)
      WRITE (6,16)
510   IF (.NOT.(PRS.OR.PRH)) GO TO 540
      WRITE (6,24)
520   IF (.NOT.PRS) GO TO 530
      WRITE (6,18) YT
      WRITE (6,25) ERR
      WRITE (6,15)
530   IF (.NOT.PRH) GO TO 540
      CALL HSTPRT (YHIST)
      WRITE (6,16)
540   IF (.NOT.PLC) GO TO 550
      CALL HSTCRV (NREC,YTAB,YHIST)
      IF (PLH) PAUSE 'TYPE C/R FOR PLOT OF C.D.F.'
      CALL CRVPLT (1,2,XTAB)
      CALL CRVPLT (2,3,YTAB)
550   WRITE (7,19)
      OLDPRH=PRH
      READ (5,3) ANS
```

```
      IF (ANS.EQ.YES) GO TO 100
      WRITE (7,26)
      READ (5,3) ANS
      IF (ANS.EQ.YES) GO TO 410
      GO TO 420
C
CC
C        THIS SECTION OF CODE IS THE HISTOGRAM FLATTENING
C        PREPROCESSOR.  THE OPTIONS AVAILABLE ARE PRINTING
C        AND PLOTTING OF HISTOGRAMS, PRINTING OF STATISTICS,
C        AND THRESHOLDING.  THE NUMBER OF ITERATIONS IS ALSO
C        SELECTABLE.  THE LABEL RECORD IS COPIED FIRST, THEN
C        THE FILES CLOSED AND REOPENED WITH THE LENGTH
C        DEFINED TO BE THE LENGTH OF ONE ROW.  THE
C        SUBROUTINE 'GRAYEQ' IS CALLED AS MANY TIMES AS
C        REQUESTED, AND THE OPTIONS ARE PERFORMED AS
C        DESIRED.  FINALLY, THE LAST HISTOGRAM IS STORED IN
C        THE OUTPUT LABEL RECORD.
C
C
  700 WRITE (7,27)
  710 WRITE (7,9)
      READ (5,3) ANS
      PLH=FALSE
      IF (ANS.EQ.YES) PLH=TRUE
      WRITE (7,13)
      READ (5,3) ANS
      PRH=FALSE
      IF (ANS.EQ.YES) PRS=TRUE
      IF ((.NOT.OLDPRH).AND.PRH) WRITE (6,16)
      WRITE (7,14)
      READ (5,3) ANS
      PRS=FALSE
      IF (ANS.EQ.YES) PRS=TRUE
      WRITE (7,38)
      READ (5,3) ANS
      TLD=FALSE
      IF (ANS.EQ.YES) TLD=TRUE
      WRITE (7,28)
      READ (5,12) ITER
      IF (ITER.LE.0) GO TO 900
  720 WRITE (7,8)
      READ (5,3) INFIL
      IF (INFIL(1).EQ.32) GO TO 9999
      CALL ASSIGN (1,INFIL,14,'OLD','NC',1)
      DEFINE FILE 1 (130,1024,U,I1)
      CALL RECNUM (LPIC,NREC,NAM1,1,ERFLG)
      IF ((NREC.EQ.0).OR.(ERFLG.NE.0)) GO TO 890
      WRITE (7,22)
      READ (5,3) OUTFIL
      IF (OUTFIL(1).EQ.32) GO TO 890
      CALL ASSIGN (2,OUTFIL,14,'NEW','NC',1)
```

```
      DEFINE FILE 2 (NREC,1024,U,I2)
      CALL RWLBLO (1,2,'%',SYSDAT,LPIC,ERFLG)
      IF (ERFLG.NE.0) GO TO 880
      CALL HIST (NREC,1,ERFLG,IPIC,XHIST)
      IF (ERFLG.NE.0) GO TO 880
      WRITE (7,23)
      CALL STAT (XT,XHIST)
      CALL UNIRMS (NREC,XHIST,ERR)
730   IF (.NOT.(PRS.OR.PRH) GO TO 750
740   WRITE (6,17) NAM1,DAT
750   IF (.NOT.PLH) GO TO 760
      CALL HSTPLT (4,1,XHIST)
760   IF (.NOT.PRS) GO TO 770
      WRITE (6,23)
      WRITE (6,18) XT
      WRITE (6,25) ERR
      WRITE (6,15)
770   IF (.NOT.PRH) GO TO 780
      CALL HSTPRT (XHIST)
      WRITE (6,16)
780   CALL WRITEO (2,NREC,IPIC,ERFLG)
      IF (ERFLG.NE.0) GO TO 880
      ENDFILE 1
      ENDFILE 2
      IF (NREC-34) 790,800,810
790   NREC1=160
      NW=64
      GO TO 820
800   NREC1=272
      NW=128
      GO TO 820
810   NREC1=520
      NW=256
820   DEFINE FILE 1 (NREC1,NW,U,I1)
      CALL ASSIGN (2,OUTFIL,14,'OLD','NC',1)
      DEFINE FILE 2 (NREC1,NW,U,I2)
      DO 870 I=1,ITER
      ERFLG=I
      CALL GRAYEQ (GRHIST,NW,XHIST,TLD,ERFLG,IPIC,LPIC)
      IF (ERFLG.NE.0) TO GO 880
      WRITE (7,39) I
      CALL STAT (XT,XHIST)
      CALL UNIRMS (NREC,XHIST,ERR)
830   IF (.NOT.PLH) GO TO 840
      K=MOD(I,4)+1
      CALL HSTPLT (4,K,XHIST)
840   IF (.NOT.(PRS.OR.PRH)) GO TO 870
850   IF (.NOT.PRS) GO TO 860
      WRITE (6,39) I
      WRITE (6,18) XT
      WRITE (6,25) ERR
      WRITE (6,15)
```

```
  860 IF (.NOT.PRH) GO TO 870
      CALL HSTPRT (XHIST)
      WRITE (6,16)
  870 CONTINUE
      ENDFILE 1
      ENDFILE 2
      CALL ASSIGN (2,OUTFIL,14,'OLD','NC',1)
      DEFINE FILE 2 (NREC,1024,U,I1)
      CALL HSTSTF (2,XHIST,ERFLG,IPIC)
      ENDFILE 2
      GO TO 900
  880 ENDFILE 2
  890 ENDFILE 1
  900 WRITE (7,19)
      OLDPRH=PRH
      READ (5,3) ANS
      IF (ANS.EQ.YES) GO TO 100
      WRITE (7,26)
      READ (5,3) ANS
      IF (ANS.EQ.YES) GO TO 710
      GO TO 720
C
C
C         THIS SECTION FUNCTIONS AS THE HISTOGRAM
C         SPECIFICATION PROCESSOR.  THE OPTIONS AVAILABLE ARE
C         PRINTING AND PLOTTING OF HISTOGRAMS, PRINTING THE
C         STATISTICS AND PLOTTING THE TRANSFORMATION
C         FUNCTION.  ALSO, IT IS POSSIBLE TO GENERATE ONLY A
C         HISTOGRAM.  THE USER HAS THE OPTION OF USING THE
C         POTENTIOMETERS OR THE JOYSTICK-CURSOR AS INPUT.
C         LABEL RECORD IS COPIED AND THE ORIGINAL HISTOGRAM
C         AND C.D.F. COMPUTED.  THE INVERSE IS TAKEN, THE
C         FUNCTIONS COMBINED AND THE PICTURE IS TRANSFORMED.
C         FINALLY THE NEW HISTOGRAM IS STORED IN THE LABEL
C         RECORD.  IF A NULL OUTPUT FILE NAME IS ENTERED, ALL
C         OF THE ABOVE WILL HAPPEN EXCEPT THAT A NEW IMAGE IS
C         NOT GENERATED.
C
C
 1000 WRITE (7,29)
 1010 WRITE (7,30)
      READ (5,3) ANS
      GNC=FALSE
      IF (ANS.EQ.YES) GNC=TRUE
      WRITE (7,9)
      READ (5,3) ANS
      PLH=FALSE
      IF (ANS.EQ.YES) PLH=TRUE
      WRITE (7,13)
      READ (5,3) ANS
      PRH=FALSE
      IF (ANS.EQ.YES) PRH=TRUE
      IF ((.NOT.OLDPRH).AND.PRH) WRITE (6,16)
```

```
      WRITE (7,14)
      READ (5,3) ANS
      PRS=FALSE
      IF (ANS.EQ.YES) PRS=TRUE
      IF (GNC) GO TO 1240
      WRITE (7,37)
      READ (5,3) ANS
      PLC=FALSE
      IF (ANS.EQ.YES) PLC=TRUE
1020  WRITE (7,31)
      READ (5,3) ANS
      POTS=TRUE
      IF (ANS.EQ.YES) POTS=FALSE
      WRITE (7,8)
      READ (5,3) INFIL
      IF (INFIL(1).EQ.32) GO TO 9999
      CALL ASSIGN (1,INFIL,14,'OLD','NC',1)
      DEFINE FILE 1 (130,1024,U,I1)
      CALL RECNUM (LPIC,NREC,NAM1,1,ERFLG)
      IF ((NREC.EQ.0).OR.(ERFLG.NE.0)) GO TO 1230
      WRITE (7,22)
      READ (5,3) OUTFIL
      OUTPUT=TRUE
      IF (OUTFIL (1).EQ.32) OUTPUT=FALSE
      IF (.NOT.OUTPUT) GO TO 1030
      CALL ASSIGN (2,OUTFIL,14,'NEW','NC',1)
      DEFINE FILE 2 (NREC,1024,U,I2)
      CALL RWLBLO (1,2,'&',SYSDAT,LPIC,ERFLG)
      IF (ERFLG.NE.0) GO TO 1220
1030  CALL HIST (NREC,1,ERFLG,IPIC,XHIST)
      IF (ERFLG.NE.0) GO TO 1220
      WRITE (7,23)
      CALL STAT (XT,XHIST)
      IF (.NOT.PLH) GO TO 1050
1040  CALL HSTPLT (4,1,XHIST)
1050  IF (.NOT.(PRS.OR.PRH)) GO TO 1080
1060  WRITE (6,17) NAM1,DAT
      WRITE (6,23)
      IF (.NOT.PRS) GO TO 1070
      WRITE (6,18) XT
      WRITE (6,15)
1070  IF (.NOT.PRH) GO TO 1080
      CALL HSTPRT (XHIST)
      WRITE (6,16)
1080  CALL HSTCRV (NREC,XTAB,XHIST)
      CALL TRHIST (XTAB,XHIST,THIST)
      WRITE (7,24)
      CALL STAT (TT,THIST)
      CALL UNIRMS (NREC,THIST,ERR)
      IF (.NOT.PLH) GO TO 1090
      CALL HSTPLT (4,2,THIST)
```

```
1090 IF (.NOT.(PRS.OR.PRH)) GO TO 1110
     WRITE (6,24)
     IF (.NOT.PRS) GO TO 1110
     WRITE (6,18) TT
     WRITE (6,25) ERR
     WRITE (6,15)
1100 IF (.NOT.PRH) GO TO 1110
     CALL HSTPRT (THIST)
     WRITE (6,16)
1110 IF (.NOT.POTS) GO TO 1120
     CALL POTENT (YHIST,ST,NREC)
     GO TO 1130
1120 CALL CAMERA (YHIST,NREC,ZHIST,THIST)
1130 WRITE (7,32)
     CALL STAT (YT,YHIST)
     IF (POTS) WRITE (7,35) ST
     IF (.NOT.PLH) GO TO 1140
     CALL HSTPLT (4,3,YHIST)
1140 IF (.NOT.(PRS.OR.PRH)) GO TO 1160
     WRITE (6,32)
     IF (.NOT.PRS) GO TO 1150
     WRITE (6,18) YT
     IF (.NOT.POTS) GO TO 1150
     WRITE (6,35) ST
1150 WRITE (6,15)
     IF (.NOT.PRH) GO TO 1160
     CALL HSTPRT (YHIST)
     WRITE (6,16)
1160 CALL HSTCRV (NREC,YTAB,YHIST)
     CALL INVCRV (YTAB,ZTAB)
     CALL GMIHX (XTAB,YTAB,ZTAB)
     CALL TRHIST (ZTAB,XHIST,ZHIST)
     WRITE (7,33)
     CALL STAT (ZT,ZHIST)
     CALL RMSERR (NREC,YHIST,ZHIST,ERR)
     IF (.NOT.PLH) GO TO 1170
     CALL HSTPLT (4,4,ZHIST)
1170 IF (.NOT.(PRS.OR.PRH)) GO TO 1190
     WRITE (6,33)
     IF (.NOT.PRS) GO TO 1180
     WRITE (6,18) ZT
     WRITE (6,34) ERR
     WRITE (6,15)
1180 IF (.NOT.PRH) GO TO 1190
     CALL HSTPRT (ZHIST)
     WRITE (6,16)
1190 IF (.NOT.OUTPUT) GO TO 1200
     WRITE (7,36)
     READ (5,3) ANS
     IF (ANS.NE.YES) GO TO 1030
     CALL XLATE (NREC,1,2,ERFLG,IPIC,ZTAB)
     IF (ERFLG.NE.0) GO TO 1220
```

```
      CALL HSTSTF (2,ZHIST,ERFLG,IPIC)
      ENDFILE 2
 1200 ENDFILE 1
      IF (.NOT.PLC) GO TO 1210
      IF (PLH) PAUSE 'TYPE C/R FOR PLOT OF CURVE'
      CALL CRVPLT (1,3,ZTAB)
 1210 WRITE (7,19)
      OLDPRH=PRH
      READ (5,3) ANS
      IF (ANS.EQ.YES) GO TO 100
      WRITE (7,26)
      READ (5,3) ANS
      IF (ANS.EQ.YES) GO TO 1010
      GO TO 1020
 1220 IF (OUTPUT) ENDFILE 2
 1230 ENDFILE 1
      GO TO 1210
 1240 K=0
 1250 WRITE (7,31)
      READ (5,3) ANS
      POTS=TRUE
      IF (ANS.EQ.YES) POTS=FALSE
      IF (.NOT.POTS) GO TO 1260
      CALL POTENT (XHIST,ST,10)
      GO TO 1270
 1260 CALL CAMERA (XHIST,10,ZHIST,THIST)
 1270 CALL STAT (XT,XHIST)
      IF (POTS) WRITE (7,35) ST
      IF (.NOT.PLH) GO TO 1290
      K=MOD(K,4)+1
 1280 CALL HSTPLT (4,K,XHIST)
 1290 IF (.NOT.(PRS.OR.PRH)) GO TO 1320
      WRITE (6,32)
 1300 IF (.NOT.PRS) GO TO 1310
      WRITE (6,18) XT
      IF (POTS) WRITE (6,35) ST
      WRITE (6,15)
 1310 IF (.NOT.PRH) GO TO 1320
      CALL HSTPRT (XHIST)
      WRITE (6,16)
 1320 WRITE (7,19)
      OLDPRH=PRH
      READ (5,3) ANS
      IF (ANS.EQ.YES) GO TO 100
      WRITE (7,26)
      READ (5,3) ANS
      IF (ANS.EQ.YES) GO TO 1010
      GO TO 1250
C
C
C        MESSAGES
C
C
```

```
    1 FORMAT (' INTERACTIVE IMAGE ENHANCEMENT SYSTEM',//)
    2 FORMAT (' HISTOGRAM EQUALIZATION?',$)
    3 FORMAT (14A1)
    4 FORMAT (' HISTOGRAM PREPROCESSOR?',$)
    5 FORMAT (' HISTOGRAM SPECIFICATION?',$)
    6 FORMAT (' UTILITY?',$)
    7 FORMAT ('0PICTURE UTILITY PROCESSOR')
    8 FORMAT (' TYPE INPUT FILE NAME -- ',$)
    9 FORMAT (' PLOT HISTOGRAM?',$)
   10 FORMAT (' ',T10,'NEW PLOT?',$)
   11 FORMAT (' ',T10,'HOW MANY PLOTS ON SCREEN (1,2,4)?',$)
   12 FORMAT (I10)
   13 FORMAT (' PRINT HISTOGRAM?',$)
   14 FORMAT (' PRINT STATISTICS?',$)
   15 FORMAT (1H0)
   16 FORMAT (1H1)
   17 FORMAT (' PICTURE NAME -- ',8A1,20X,'DATE:  ',9A1)
   18 FORMAT (' M1 = ',E11.4,' M2 = ',E11.4,' M3 = ',E11.4,
     1' M4 = ',E11.4,/,' MODE = ',F8.1,' AT ',F5.1,
     2' MEDIAN = ',F5.1,' MIN = ',F5.1,' MAX = ',F5.1,
     3' A3 = ',E11.4,' A4 = ',E11.4)
   19 FORMAT (' SWITCH MODES?',$)
   20 FORMAT ('0HISTOGRAM EQUALIZATION PROCESSOR')
   21 FORMAT (' PLOT CUMULATIVE DISTRIBUTION FUNCTION?',$)
   22 FORMAT (' TYPE OUTPUT FILE NAME -- ',$)
   23 FORMAT (' ORIGINAL IMAGE')
   24 FORMAT (' HISTOGRAM EQUALIZED IMAGE')
   25 FORMAT (' RMS ERROR WITH RESPECT TO UNIFORM = ',E11.4)
   26 FORMAT (' CHANGE PARAMETERS?',$)
   27 FORMAT ('0HISTOGRAM FLATTENING PREPROCESSOR')
   28 FORMAT (' ITERATION COUNT?',$)
   29 FORMAT ('0DIRECT HISTOGRAM SPECIFICATION PROCESSOR')
   30 FORMAT (' GENERATE HISTOGRAM ONLY?',$)
   31 FORMAT (' INPUT FROM JOYSTICK-CURSOR?',$)
   32 FORMAT (' DESIRED IMAGE')
   33 FORMAT (' RESULTING IMAGE')
   34 FORMAT (' RMS ERROR WITH RESPECT TO DESIRED = ',E11.4)
   35 FORMAT (' M = ',E11.4,' LS = ',E11.4,' RS = ',E11.4,
     1' H = ',E11.4)
   36 FORMAT (' SATISFIED?',$)
   37 FORMAT (' PLOT TRANSFORMATION CURVE?',$)
   38 FORMAT (' THRESHOLD?',$)
   39 FORMAT (' ITERATION #',I4)
 9999 STOP 'IIES FINISHED'
      END
```

```
      SUBROUTINE CAMERA (HISTO,NREC,XT,YT)
C
C
C        THIS SUBROUTINE GENERATES A HISTOGRAM 'HISTO' THAT
C        IS ENTERED VIA THE DIGITIZING CAMERA USING THE
C        JOYSTICK-CURSOR TO TRACE THE CURVE.  THE HISTOGRAM
C        IS SCALED TO THE CORRECT NUMBER OF POINTS USING
C        'NREC', THE NUMBER OF 2048 BYTE RECORDS IN THE
C        FILE.  'XT' AND 'YT' ARE USED AS WORK SPACE.
C
C
C        THE USER FIRST POSITIONS THE CURSOR ALONG THE ZERO
C        LINE OF THE HISTOGRAM.  THIS VALUE IS ENTERED AS
C        THE ZERO Y VALUE.  THEN THE MAXIMUM X VALUE IS
C        ESTABLISHED.  THE USER HAS THE OPTION OF DIGITIZING
C        CONTINUOUSLY OR PIECE-WISE.  IF PIECE-WISE, EACH
C        POINT IS SAMPLED BY TYPING A CARRIAGE RETURN.  THE
C        X-Y COORDINATES ARE SAMPLED AND STORED IN 'XT' AND
C        'YT' RESPECTIVELY.  ANY POINT WHERE Y IS LESS THAN
C        THE ZERO REFERENCE ARE IGNORED.  ANY POINTS WHERE
C        X IS LESS THAN A PREVIOUS VALUE ARE ALSO DISCARDED.
C        WHEN X REACHES XMAX, THE DIGITIZATION IS COMPLETE.
C        THE X VALUES ARE THEN RESCALED TO (1,256).  THE
C        Y VALUES ARE THEN USED AS BREAKPOINTS TO GENERATE
C        A PIECE-WISE LINEAR HISTOGRAM.  IF MORE THAN ONE
C        ELEMENT IN X HAS THE SAME VALUE, THE FIRST ONE IS
C        USED.  THE REST ARE DISCARDED.  THE HISTOGRAM IS
C        THEN RESCALED TO THE CORRECT NUMBER OF POINTS.
C
C        SUBROUTINES USED - CURSOR
C
C
      INTEGER*2 XT(512),YT(512),XZ,XM,YZ
      REAL*4 HISTO(256)
      LOGICAL*1 ANS,Y
      DATA Y/'Y'/
  100 PAUSE 'ENTER ZERO Y'
      CALL CURSOR (IX,YZ)
      YZ=511-YZ
      PAUSE 'ENTER MAXIMUM X'
      CALL CURSOR (XM,IY)
      WRITE (7,1)
      READ (5,2) ANS
      PAUSE 'POSITION CURSOR AT LEFT-MOST POINT AND TYPE CR'
      I=1
      IXT=-1
  110 IF (ANS.NE.Y) PAUSE 'TYPE C/R'
      CALL CURSOR (IX,IY)
      IF (IX.GT.XM) GO TO 120
      IY=511-IY
      IF ((IY.LT.YZ).OR.(IX.LE.IXT)) GO TO 110
      IXT=IX
```

```
      XT(I)=IX
      YT(I)=IY-YZ
      I=I+1
      IF (I.LE.512) GO TO 110
  120 I=I-1
      WRITE (7,3) I
      XM=XT(I)
      XZ=XT(1)
      DO 130 J=1,I
      XT(J)=(XT(J)-XZ)*255.0/(XM-XZ)+1.5
  130 CONTINUE
      K=1
      J=1
      IXT=-1
  140 IX=XT(K)
      IF (IXT.EQ.IX) GO TO 170
      IF (IX.NE.J) GO TO 150
      HISTO(J)=YT(K)
      J=J+1
      GO TO 170
  150 FYD=YT(K)-HISTO(J-1)
      IXD=IX-J
      DO 160 M=1,IXD
      HISTO(J)=HISTO(J-1)+FYD/IXD
      J=J+1
  160 CONTINUE
  170 IXT=IX
      K=K+1
      IF (K.LE.I) GO TO 140
      T=0.0
      DO 180 I=1,256
      T=T+HISTO(I)
  180 CONTINUE
      FH=(NREC-2)*2048.0/T
      DO 190 I=1,256
      HISTO(I)=HISTO(I)*FH
  190 CONTINUE
    1 FORMAT (' CONTINUOUS OR PIECE-WISE',
     1' (Y FOR CONTINUOUS)?',$)
    2 FORMAT (A1)
    3 FORMAT (' DIGITIZATION COMPLETE IN ',I4,' POINTS')
      RETURN
      END


      SUBROUTINE CRVGEN (HIST,IA,IB,IC,AA,BB,CC)
C
C
C        THIS SUBROUTINE GENERATES THE PIECE-WISE LINEAR
C        HISTOGRAM 'HIST'.  'IA', 'IB' AND 'IC' ARE THE
C        X COORDINATES OF THE THREE BREAKPOINTS.  'AA',
```

```
C          'BB' AND 'CC' ARE THE Y COORDINATES.  THE X
C          COORDINATES ARE ON THE RANGE (1,256).  THE
C          HISTOGRAM IS PIECE-WISE LINEAR ON THE FOLLOWING
C          SEGMENTS:  (1,0) - (IA,AA) - (IB,BB) - (IC,CC) -
C          (256,0).  IT COULD BE THAT 'IA'=1, OR 'IC'=256.
C          IT CHECKS FOR THIS AND TAKES THE APPROPRIATE
C          ACTION.
C
C          SUBROUTINES USED - NONE
C
C
      REAL*4 HIST(256)
      IF (IA.EQ.1) GO TO 110
      DO 100 I=1,IA
      HIST (I)=(I-1)*AA/(IA-1)
  100 CONTINUE
      GO TO 120
  110 HIST(1)=AA
  120 K=IA+1
      IF (IA.NE.IB) GO TO 130
      HIST (IB)=BB
      GO TO 150
  130 DO 140 I=K,IB
      HIST (I)=HIST(IA)+(I+1-K)*(BB-AA)/(IB-IA)
  140 CONTINUE
  150 K=IB+1
      IF (IB.EQ.IC) GO TO 170
      DO 160 I=K,IC
      HIST(I)=HIST(IB)+(I+1-K)*(CC-BB)/(IC-IB)
  160 CONTINUE
  170 IF (IC.EQ.256) GO TO 190
      HIST(IC)=CC
      K=IC+1
      DO 180 I=K,256
      HIST(I)=HIST(IC)+(I+1-K)*(-CC)/(256-IC)
  180 CONTINUE
  190 RETURN
      END


      SUBROUTINE CRVPLT (M,REF,TABLE)
C
C
C          THIS SUBROUTINE PLOTS A TRANSFORMATION CURVE PASSED
C          TO IT IN 'TABLE'.  IT ASSUMES THAT THE VALUES IN
C          'TABLE' ARE IN THE RANGE (0,255).  'M' INDICATES
C          WHETHER OR NOT TO PLOT A GRID AND CLEAR THE SCREEN.
C          IF M IS 1, IT IS DONE.  IF NOT, THE CURVE IS
C          PLOTTED OVER WHATEVER IS THERE.  A 45 DEGREE
C          REFERENCE LINE IS PLOTTED IF 'REF'=2 AND 'M'=1
C          THIS SUBROUTINE IS SET UP SO THAT IF A HISTOGRAM IS
```

```
C          PLOTTED USING 'HSTPLT', WITH 'NUM'=1, 'PASS'=1,
CC         THE C.D.F. CAN BE PLOTTED OVER IT WITH A CALL TO
C          'CRVPLT' WITH 'M' NOT EQUAL TO 1.
C
C          SUBROUTINES USED - NEWPEN,PLOTS,VIDPLT
C
C
      INTEGER*2 TABLE(256),REF
      IF (M.NE.1) GO TO 120
  100 CALL PLOTS (0)
      CALL VIDPLT (17,13,-3)
      CALL NEWPEN (6)
      DO 110 I=1,9
      IY=50*(I-1)
      CALL VIDPLT (0,IY,3)
      CALL VIDPLT (512,IY,2)
      IX=64*(I-1)
      CALL VIDPLT (IX,0,3)
      CALL VIDPLT (IX,400,2)
  110 CONTINUE
      REF=IABS(REF)
      CALL VIDPLT (0,0,REF)
      CALL NEWPEN (12)
  120 CALL VIDPLT (0,0,3)
      DO 130 I=1,256
      IY=1.5625*TABLE(I)+0.5
      IX=2*(I-1)
      CALL VIDPLT (IX,IY,2)
  130 CONTINUE
      RETURN
      END


      SUBROUTINE DIFF (DELT,IA,IB,IM,TLD,IC,J)
C
CC         THIS SUBROUTINE COMPUTES THE DIFFERENCE SUMMATION
C          'DELT' IF 'IC'=1.  'IA' IS THE ORIGINAL POINT,
C          'IB' IS THE 9 POINT SUMMATION TO REPLACE 'IA'.
C          IF 'TLD' IS FALSE, THRESHOLDING IS NOT DESIRED SO
C          IT RETURNS.  IF 'IC'=2, THE DIFFERENCE IS COMPUTED
C          AND TESTED TO SEE IF IT LIES WITHIN THE BOUNDARY
C          AROUND 9*IA.  IF NOT, 'J' IS USED TO REASSIGN THE
C          POINT TO SOME VALUE WITHIN THE LIMITS.  'IM' IS THE
C          MEAN OF THE SUMMATION 'DELT' IF 'IC'=2.
C
C          SUBROUTINES USED - NONE
C
C
      REAL*4 DELT(2)
      LOGICAL TLD
```

```
      IF (.NOT.TLD) RETURN
      ID=9*IA
      I=IABS(ID-IB)
      IF (I.EQ.0) RETURN
      GO TO (100,110),IC
  100 DELT(1)=DELT(1)+I
      DELT(2)=DELT(2)+1.0
      RETURN
  110 IF (I.LE.IM) RETURN
      IS=ISIGN(1,(IB-ID))
      J=MOD(J,IM+1)+1
      IB=ID+J*IS
      RETURN
      END


      SUBROUTINE EQUAL (NW,HIST,DELT,IM)
C
C
C        THIS SUBROUTINE COMPUTES THE CUMULATIVE
C        DISTRIBUTION FUNCTION OF THE HISTOGRAM 'HIST' AND
C        STORES IT BACK IN 'HIST' WITH VALUES ON THE RANGE
C        (0,255).  IT ALSO COMPUTES THE MEAN OF THE
C        DIFFERENCE SUMMATION 'DELT' AND RETURNS IT IN 'IM'.
C        'NW' IS THE NUMBER OF POINTS IN A ROW OF THE PICTURE.
C
C        SUBROUTINES USED - NONE
C
C
      REAL*4 HIST(2304),DELT(2)
      IM=0
      WORD=FLOAT(NW)*FLOAT(NW)
      SCALE=255.0/WORD
      T=0.0
      DO 100 I=1,2304
      T=T+HIST(I)
      HIST(I)=AINT(T*SCALE+0.5)-128.0
      IF (HIST(I).LT.0.0) HIST(I)=HIST(I)+256.0
  100 CONTINUE
      IF (DELT(2).EQ.0.0) RETURN
      IM=DELT(1)/DELT(2)+0.5
      RETURN
      END


      SUBROUTINE FIX (IC,NREC,HIST,OPIC,LOPIC,NW,HIST1,ERFLG)
C
C
C        IF 'IC'=1, THIS SUBROUTINE UPDATES THE HISTOGRAM
C        'HIST' USING THE DATA IN 'OPIC'.  'NW' IS THE
```

```
C          NUMBER OF DATA POINTS IN 'OPIC'.  'LOPIC' IS THE
C          SAME ARRAY AS 'OPIC', BUT IN LOGICAL*1 RATHER THAN
C          INTEGER*2.  IF 'IC'=2, THIS SUBROUTINE TRANSLATES
C          THE DATA IN 'OPIC' USING 'HIST' AS THE
C          TRANSFORMATION.  IT ALSO COMPUTES THE 256 POINT
C          HISTOGRAM 'HIST1'.  'OPIC' IS THEN OUTPUT TO RECORD
C          'NREC' IN THE OUTPUT FILE.  'ERFLG'=0 IF THERE WERE
C          NO ERRORS.
C
C          SUBROUTINES USED - NONE
C
C
      INTEGER*2 OPIC(512),ERFLG
      LOGICAL*1 LOPIC(1024)
      REAL*4 HIST(2304),HIST1(256)
      ERFLG=0
      NB=NW+NW
      GO TO (100,120),IC
  100 DO 110 I=1,NW
      K=OPIC(I)+1
      HIST(K)=HIST(K)+1.0
  110 CONTINUE
      RETURN
  120 DO 130 I=1,NW
      K=OPIC(I)+1
      OPIC(I)=HIST(K)
      M=HIST(K)
      HIST1(M+1)=HIST1(M+1)+1.0
  130 CONTINUE
      WRITE (2'NREC,END=140,ERR=150) (LOPIC(I),I=1,NB,2)
      RETURN
  140 WRITE (7,1)
      GO TO 160
  150 WRITE (7,2)
  160 ERFLG=1
    1 FORMAT (' END OF FILE ON UNIT 2 IN ''FIX''')
    2 FORMAT (' I/O ERROR ON UNIT 2 IN ''FIX''')
      RETURN
      END


      SUBROUTINE GM1HX (H,G,T)
C
C
C          THIS SUBROUTINE CREATES THE TRANSFORMATION
C          T=G(H(X)).
C
C          SUBROUTINES USED - NONE
C
C
      INTEGER*2 H(256),G(256),T(256)
      DO 100 I=1,256
```

```
      J=H(I)+1
      T(I)=G(J)
  100 CONTINUE
      RETURN
      END


      SUBROUTINE GRAYEQ (HIST,NX,HIST1,TLD,ERFLG,IPIC,LPIC)
C
C
C        THIS SUBROUTINE PERFORMS THE HISTOGRAM FLATTENING
C        OPERATION.  TWO COMPLETE PASSES ARE MADE THROUGH
C        THE PICTURE.  ON THE FIRST PASS, THE 2,304 POINT
C        HISTOGRAM 'HIST' IS COMPUTED, AS WELL AS THE
C        DIFFERENCE SUMMATION 'DELT'.  THE C.D.F. OF 'HIST'
C        IS THEN COMPUTED AND STORED BACK INTO 'HIST'.  THE
C        SECOND PASS THEN USES 'HIST' TO TRANSLATE THE DATA
C        AND THRESHOLD IT IF 'TLD' IS TRUE.  'HIST1' IS THE
C        256 POINT HISTOGRAM OF THE RESULTING IMAGE.
C        'ERFLG'=0 IF THERE WERE NO ERRORS.  'IPIC' IS USED
C        AS WORKSPACE.  "LPIC' IS THE SAME ARRAY BUT IN
C        LOGICAL*1 FORM.  'NX' IS ONE HALF THE NUMBER OF
C        POINTS IN ONE ROW.  THE SUBROUTINE OPERATES ON 3
C        ROWS AT A TIME TO COMPUTE A 9 POINT SUMMATION.
C        THEY ARE STORED IN THE FIRST 3 ROWS OF 'IPIC'.
C        THE TOP AND BOTTOM ROWS AND ALONG THE SIDES ARE
C        ONLY 6 POINT SUMMATIONS, BUT THEY ARE MULTIPLIED
C        BY 1.5.
C        THE FOUR CORNERS ARE 4 POINT SUMMATIONS MULTIPLIED
C        BY 2.25.  THE TOP AND BOTTOM ROWS ARE PROCESSED
C        FIRST, THEN THE OTHERS.  THE I/O IS SET UP SO THAT
C        ONLY 1 ROW NEED BE READ AT A TIME, RATHER THAN ALL
C        THREE.  IT ASSUMES INPUT ON LUN 1, OUTPUT ON LUN 2.
C
C        SUBROUTINES USED - DIFF,EQUAL,FIX,JUG
C
C
      INTEGER*2 IPIC(512,4),OPIC(512),ERFLG
      LOGICAL*1 LPIC(1024,4),LOPIC(1024)
      REAL*4 HIST(2304),DELT(2),HIST1(256)
      LOGICAL TLD
      EQUIVALENCE (OPIC(1),LOPIC(1))
      L=1
      IF (ERFLG.NE.1) L=2
      ERFLG=0
      NR=2048/NX
      NW=NX+NX
      NB=NW+NW
      IEN=NR+NW
      IEN1=IEN-1
      NW1=NW-1
```

```
      DO 100 I=1,2304
      HIST(I)=0.0
  100 CONTINUE
      DELT(1)=0.0
      DELT(2)=0.0
      DO 110 I=1,256
      HIST1(I)=0.0
  110 CONTINUE
      DO 120 J=1,3
      DO 120 I=1,NW
      IPIC(I,J)=0
  120 CONTINUE
      DO 180 IC=1,2
      READ (L'NR+1,END=190,ERR=200)  (LPIC(J,1),J=1,NB,2)
      READ (L'NR+2,END=190,ERR=200)  (LPIC(J,2),J=1,NB,2)
      CALL JUG (IPIC,NW,1)
      CALL JUG (IPIC,NW,2)
      OPIC(1)=(IPIC(1,1)+IPIC(1,2)+IPIC(2,1)+IPIC(2,2))*2.25
      IJK=0
      CALL DIFF (DELT,IPIC(1,1),OPIC(1),IM,TLD,IC,IJK)
      OPIC(NW)=(IPIC(NW,1)+IPIC(NW1,1)+IPIC(NW,2)+
     1IPIC(NW1,2))*2.25
      CALL DIFF (DELT,IPIC(NW,1),OPIC(NW),IM,TLD,IC,IJK)
      DO 130 J=2,NW1
      OPIC(J)=(IPIC(J-1,1)+IPIC(J,1)+IPIC(J+1,1)+IPIC(J-1,2)+
     1IPIC(J,2)+IPIC(J+1,2))*1.5+0.5
      CALL DIFF (DELT,IPIC(J,1),OPIC(J),IM,TLD,IC,IJK
  130 CONTINUE
      CALL FIX (IC,NR+1,HIST,OPIC,LOPIC,NW,HIST1,ERFLG)
      IF (ERFLG.NE.0) RETURN
      READ (L'IEN1,END=190,ERR=200)  (LPIC(J,1),J=1,NB,2)
      READ (L'IEN,END=190,ERR=200)  (LPIC(J,2),J=1,NB,2)
      CALL JUG (IPIC,NW,1)
      CALL JUG (IPIC,NW,2)
      OPIC(1)=(IPIC(1,1)+IPIC(1,2)+IPIC(2,1)+IPIC(2,2))*2.25
      CALL DIFF (DELT,IPIC(1,2),OPIC(1),IM,TLD,IC,IJK)
      OPIC(NW)=(IPIC(NW,1)+IPIC(NW1,1)+IPIC(NW,2)+
     1IPIC(NW1,2))*2.25
      CALL DIFF (DELT,IPIC(NW,2),OPIC(NW),IM,TLD,IC,IJK)
      DO 140 J=2,NW1
      OPIC(J)=(IPIC(J-1,1)+IPIC(J,1)+IPIC(J+1,1)+IPIC(J-1,2)+
     1IPIC(J,2)+IPIC(J+1,2))*1.5+0.5
      CALL DIFF (DELT,IPIC(J,2),OPIC(J),IM,TLD,IC,IJK)
  140 CONTINUE
      CALL FIX (IC,IEN,HIST,OPIC,LOPIC,NW,HIST1,ERFLG)
      IF (ERFLG.NE.0) RETURN
      K=NR+2
      READ (L'K-1,END=190,ERR=200)  (LPIC(J,1),J=1,NB,2)
      READ (L'K,END=190,ERR=200)  (LPIC(J,2),J=1,NB,2)
      CALL JUG (IPIC,NW,1)
      CALL JUG (IPIC,NW,2)
      IK=2
```

```
      DO 170 M1=3,NW
      IROW=MOD(IK,3)+1
      READ (L'K+1,END=190,ERR=200)  (LPIC(J,IROW),J=1,NB,2)
      IK=IK+1
      IROW1=MOD(IROW+1,3)+1
      CALL JUG (IPIC,NW,IROW)
      OPIC(1)=(IPIC(1,1)+IPIC(1,2)+IPIC(1,3)+IPIC(2,1)+
     1IPIC(2,2)+IPIC(2,3))*1.5+0.5
      CALL DIFF (DELT,IPIC(1,IROW1),OPIC(1),IM,TLD,IC,IJK)
      OPIC(NW)=(IPIC(NW1,1)+IPIC(NW1,2)+IPIC(NW1,3)+
     1IPIC(NW,1)+IPIC(NW,2)+IPIC(NW,3))*1.5+0.5
      CALL DIFF (DELT,IPIC(NW,IROW1),OPIC(NW),IM,TLD,IC,IJK)
      DO 160 M2=2,NW1
      OPIC(M2)=0
      DO 150 M3=1,3
      DO 150 M4=1,3
      K1=M2+M3-2
      OPIC(M2)=OPIC(M2)+IPIC(K1,M4)
  150 CONTINUE
      CALL DIFF (DELT,IPIC(M2,IROW1),OPIC(M2),IM,TLD,IC,IJK)
  160 CONTINUE
      CALL FIX (IC,K,HIST,OPIC,LOPIC,NW,HIST1,ERFLG)
      IF (ERFLG.NE.0) RETURN
      K=K+1
  170 CONTINUE
      IF (IC.EQ.2) GO TO 180
      CALL EQUAL (NW,HIST,DELT,IM)
  180 CONTINUE
      RETURN
  190 WRITE (7,1)
      GO TO 210
  200 WRITE (7,2)
  210 ERFLG=1
    1 FORMAT (' END OF FILE ON UNIT 1 IN ''GRAYEQ''')
    2 FORMAT (' I/O ERROR ON UNIT 1 IN ''GRAYEQ''')
      RETURN
      END


      SUBROUTINE HIST (NREC,LUN,ERFLG,IPIC,HISTO)
C
C
C        THIS SUBROUTINE COMPUTES THE HISTOGRAM OF THE
C        PICTURE IN FILE 'LUN'.  THE HISTOGRAM IS STORED IN
C        'HISTO'.  IT FIRST CHECKS TO SEE IF THE HISTOGRAM
C        HAS BEEN STORED IN THE LABEL RECORD.  IF IT HAS,
C        IT IS MOVED INTO 'HISTO'.  IF IT HAS NOT, IT IS
C        COMPUTED AND THEN WRITTEN INTO THE LABEL RECORD.
C        'IPIC' IS USED AS AN I/O BUFFER.  'ERFLG' IS ZERO
C        IF THERE WERE NO ERRORS.  'NREC' IS THE NUMBER OF
C        RECORDS IN THE FILE.
```

```
C
C          SUBROUTINES USED - READO
C
C
      INTEGER*2 IPIC(2048),ERFLG,IHIST(512)
      REAL HISTO(256),TEMP(256)
      EQUIVALENCE (IHIST(1),TEMP(1))
      ERFLG=0
      READ (LUN'1,END=170,ERR=180) (IPIC(I),I=1,1024)
      IF (IPIC(512).NE.-1) GO TO 120
      DO 100 I=1,512
      K=512+I
      IHIST(I)=IPIC(K)
  100 CONTINUE
      DO 110 I=1,256
      HISTO(I)=TEMP(I)
  110 CONTINUE
      RETURN
  120 DO 130 I=1,256
      HISTO(I)=0.0
  130 CONTINUE
      DO 140 I=3,NREC
      CALL READO (LUN,I,ERFLG,IPIC)
      IF (ERFLG.NE.0) GO TO 999
      DO 140 J=1,2048
      M=IPIC(J)+1
      HISTO(M)=HISTO(M)+1.0
  140 CONTINUE
      READ (LUN'1,END=170,ERR=180) (IPIC(I),I=1,1024)
      IPIC(512)=-1
      DO 150 I=1,256
      TEMP(I)=HISTO(I)
  150 CONTINUE
      DO 160 I=1,512
      K=512+I
      IPIC(K)=IHIST(I)
  160 CONTINUE
      WRITE (LUN'1,END=170,ERR=180) (IPIC(I),I=1,1024)
      RETURN
  170 WRITE (7,1) LUN
      GO TO 190
  180 WRITE (7,2) LUN
  190 ERFLG=1
    1 FORMAT (' END OF FILE ON UNIT ',I3,' IN ''HIST''')
    2 FORMAT (' I/O ERROR ON UNIT ',I3,' IN ''HIST''')
  999 RETURN
      END
```

```
      SUBROUTINE HSTCRV (NREC,TABLE,HISTO)
C
C
C        THIS SUBROUTINE COMPUTES THE CUMULATIVE
C        DISTRIBUTION FUNCTION 'TABLE' OF THE HISTOGRAM
C        'HISTO'.  IT IS SCALED ON THE RANGE (0,255) SO THAT
C        IT MAY BE USED AS A TRANSFORMATION TABLE FOR
C        HISTOGRAM EQUALIZATION.  'NREC' DENOTES THE NUMBER
C        OF 2048 BYTE RECORDS IN THE FILE.
C
C        SUBROUTINES USED - NONE
C
C
      INTEGER*2 TABLE (256)
      REAL HISTO(256)
      PIXEL=2048.0*(NREC-2)
      SCALE=255.0/PIXEL
      TEMP=0.0
      DO 100 I=1,255
      TEMP=SCALE*HISTO(I)+TEMP
      TABLE(I)=TEMP+0.5
  100 CONTINUE
      TABLE(256)=255
      RETURN
      END


      SUBROUTINE HSTPLT (NUM,PASS,HISTO)
C
C
C        THIS SUBROUTINE PLOTS A SERIES OF HISTOGRAMS ON THE
C        TELEVISION.  'NUM' INDICATES THE NUMBER OF
C        HISTOGRAMS TO BE PLOTTED ON THE SCREEN AT ONCE -
C        EITHER 1, 2, OR 4.  'PASS' INDICATES WHICH
C        HISTOGRAM IS CURRENTLY BEING PLOTTED.  'HISTO' IS
C        THE HISTOGRAM TO BE PLOTTED.  IF 'PASS'=1, THE
C        SCREEN IS CLEARED AND THE GRID PLOTTED.  SUBSEQUENT
C        PASSES MERELY MOVE THE ORIGIN UP THE LEFT HAND
C        VERTICAL AXIS.
C
C        SUBROUTINES USED - NEWPEN,PLOTS,VIDPLT
C
C
      INTEGER*2 NUM,PASS
      REAL*4 HISTO(256)
      IF (NUM.EQ.3) NUM=4
      INC=400/NUM
      IF (PASS.NE.1) GO TO 100
      CALL PLOTS (0)
  100 CALL NEWPEN (6)
      IF (PASS.NE.1) GO TO 120
```

```
      CALL VIDPLT (17,3,-3)
      DO 110 I=1,5
      IY=100*(I-1)
      CALL VIDPLT (0,IY,3)
      CALL VIDPLT (515,IY,2)
      CALL VIDPLT (515,IY+1,2)
      CALL VIDPLT (0,IY+1,2)
  110 CONTINUE
      CALL VIDPLT (0,0,2)
      CALL VIDPLT (1,0,2)
      CALL VIDPLT (1,401,2)
      CALL VIDPLT (514,0,3)
      CALL VIDPLT (514,401,2)
      CALL VIDPLT (515,401,2)
      CALL VIDPLT (515,0,2)
      CALL VIDPLT (2,1,-3)
  120 IF (PASS.EQ.1) GO TO 130
      CALL VIDPLT (0,INC,-3)
  130 DO 140 I=1,3
      IX=128*I
      CALL VIDPLT (IX,3,3)
      CALL VIDPLT (IX,-3,2)
      CALL VIDPLT (IX+1,-3,2)
      CALL VIDPLT (IX+1,3,2)
  140 CONTINUE
      CALL VIDPLT (0,0,3)
      FM=0.0
      DO 150 I=1,256
      IF (FM.LT.HISTO(I)) FM=HISTO(I)
  150 CONTINUE
      SCALE=(INC-1)/FM
      CALL NEWPEN (9)
      DO 160 I=1,256
      IF (HISTO(I).EQ.0.0) GO TO 160
      IX=2*(I-1)
      CALL VIDPLT (IX,0,3)
      IY=HISTO(I)*SCALE+0.5
      CALL VIDPLT (IX,IY,2)
  160 CONTINUE
      RETURN
      END


      SUBROUTINE HSTPRT (HIST)
C
C
C        THIS SUBROUTINE TAKES A 256 POINT HISTOGRAM 'HIST'
C        AND PLOTS IT ON THE LINE PRINTER.
C
C        SUBROUTINES USED - NONE
C
C
```

```
      REAL*4 HIST(256)
      LOGICAL*1 LINE(101)
      DATA LINE/101*'*'/
      T=0.0
      DO 100 I=1,256
      T=AMAX1(T,HIST(I))
  100 CONTINUE
      T=100.0/T
      DO 110 I=1,256
      J=T*HIST(I)+1.5
      K=I-1
      WRITE (6,1) K,HIST(I),(LINE(M),M=1,J)
  110 CONTINUE
    1 FORMAT (1H ,I3,2X,F8.1,1X,101A1)
      RETURN
      END


      SUBROUTINE HSTSTF (LUN,HIST,ERFLG,IPIC)
C
C
C        THIS SUBROUTINE PUTS THE HISTOGRAM 'HIST' INTO THE
C        LABEL RECORD OF THE PICTURE IN 'LUN'.  IT ALSO SETS
C        THE FLAG THAT INDICATES THAT IT IS THERE.  'IPIC'
C        IS USED FOR AN I/O BUFFER.  'ERFLG' IS SET TO ZERO
C        IF THERE ARE NO ERRORS.
C
C        SUBROUTINES USED - NONE
C
C
      REAL*4 HIST(256),FNUM
      INTEGER*2 ERFLG,IPIC(2048),INUM(2)
      EQUIVALENCE (FNUM,INUM(1))
      ERFLG=0
      READ (LUN'1,END=110,ERR=120) (IPIC(I),I=1,1024)
      IPIC(512)=-1
      DO 100 I=2,512,2
      K=I/2
      FNUM=HIST(K)
      IPIC(512+I-1)=INUM(1)
      IPIC(512+I)=INUM(2)
  100 CONTINUE
      WRITE (LUN'1,END=110,ERR=120) (IPIC(I),I=1,1024)
      RETURN
  110 WRITE (7,1) LUN
      GO TO 130
  120 WRITE (7,2) LUN
  130 ERFLG=1
    1 FORMAT (' END OF FILE ON UNIT ',I3,' IN ''HSTSTF''')
    2 FORMAT (' I/O ERROR ON UNIT ',I3,' IN ''HSTSTF''')
      RETURN
      END
```

```
      SUBROUTINE INPUT (ST)
C
C
C        THIS SUBROUTINE INPUTS THE DATA FROM THE FOUR
C        POTENTIOMETERS.  IT THEN MANIPULATES THEM SO THAT:
C        ST(1) IS A VALUE FOR M ON THE RANGE (0,1); ST (2)
C        IS THE LEFT SPREAD - IF IT IS POSITIVE, IT IS A
C        DISPLACEMENT, NEGATIVE IS AN ANGLE (0,90); ST(3)
C        IS THE RIGHT SPREAD - SAME EXPLANATION; ST(4) IS
C        THE HEIGHT,H.
C
C        SUBROUTINES USED - JOYSTK
C
C
      INTEGER*2 A,B,C,D
      REAL*4 ST(4)
      PAUSE 'TYPE C/R TO SET POTS'
      CALL JOYSTK (A,B,C,D)
      ST(1)=A/4096.0+0.5
      ST(2)=((90.0*B)/2048.0)
      IF (B.GT.0) ST(2)=B*0.01
      ST(3)=((90.0*C)/2048.0)
      IF (C.GT.0) ST(3)=C*0.01
      ST(4)=1.0+(D/2048.0)
      IF (D.GT.0) ST(4)=1.0+D*0.01
      RETURN
      END


      SUBROUTINE INVCRV (TABLE,TEMP)
C
C
C        THIS SUBROUTINE TAKES A TRANSFORMATION 'TABLE',
C        COMPUTES THE INVERSE OF THE TRANSFORMATION AND
C        STORES IT BACK INTO 'TABLE', USING 'TEMP' FOR
C        WORK SPACE.
C        THE INDEPENDENT VARIABLE IS ASSUMED TO HAVE VALUES
C        ON THE RANGE (0,255).  THE DEPENDENT VARIABLE IS
C        ASSUMED TO HAVE THE SAME INTEGER RANGE.  IF
C        SEVERAL VALUES OF THE INDEPENDENT VARIABLE ARE
C        MAPPED TO THE SAME VALUE OF DEPENDENT VARIABLE,
C        THE INVERSE IS COMPUTED SUCH THAT THE DEPENDENT
C        VALUE IS MAPPED BACK TO ITS FIRST OCCURRENCE.  IF
C        THERE ARE NO OCCURRENCES OF A CERTAIN DEPENDENT
C        VALUE IN THE INDEPENDENT VARIABLE, THE DEPENDENT
C        VALUE WILL MAP TO THE INDEPENDENT VARIABLE WHOSE
C        DEPENDENT VALUE IS CLOSEST TO IT.
C
C        SUBROUTINES USED - NONE
C
C
```

```
      INTEGER*2 TABLE(256),TEMP(256)
      J=1
      K=TABLE(1)+1
      DO 100 I=1,K
      TEMP(I)=0
  100 CONTINUE
      M=K+1
      DO 140 I=M,256
      K=I-1
  110 IF (K.EQ.TABLE(J)) GO TO 130
      IF (K.LT.TABLE(J)) GO TO 120
      J=J+1
      GO TO 110
  120 IF ((K-TABLE(J-1)).LT.(TABLE(J)-K)) J=J-1
  130 TEMP(I)=J-1
  140 CONTINUE
      DO 150 I=1,256
      TABLE(I)=TEMP(I)
  150 CONTINUE
      RETURN
      END


      SUBROUTINE JUG (IPIC,NW,J)
C
C
C        THIS SUBROUTINE CONVERTS 'NW' ELEMENTS IN ROW 'J'
C        OF 'IPIC' FROM BYTE DATA ON THE INTERVAL
C        (-128,+127) TO INTEGER DATA ON THE RANGE (0,255).
C
C        SUBROUTINES USED - NONE
C
C
      INTEGER*2 IPIC(512,4)
      DO 100 I=1,NW
      IF (IPIC(I,J).GE.128) IPIC(I,J)=IPIC(I,J)-256
      IPIC(I,J)=IPIC(I,J)+128
  100 CONTINUE
      RETURN
      END


      SUBROUTINE POTENT (HIST,ST,NREC)
C
C
C        THIS SUBROUTINE CALLS 'INPUT' WHICH GETS THE
C        COEFFICIENTS 'ST' FROM THE POTENTIOMETERS.
C        IF LS, RS AND H ARE ZERO, AND M=0.5, IT GENERATES
C        A UNIFORM HISTOGRAM AND RETURNS.  IF LS, RS AND H
C        ARE ZERO AND M IS NOT 0.5, IT GENERATES A
C        HISTOGRAM THAT IS FLAT ON AN INTERVAL SYMMETRIC
```

```
C          ABOUT M, ZERO ELSEWHERE.  IN ANY OTHER CASE, IT
C          COMPUTES THE BREAKPOINTS OF THE PIECE-WISE LINEAR
C          HISTOGRAM.  'H' IS THE Y COORDINATE AT 'M'.  IF
C          'LS' AND/OR 'RS' ARE LESS THAN ZERO, THEY
C          REPRESENT ANGLES.  THE X AND Y COORDINATES ARE
C          THEN COMPUTED USING THE TRIGONOMETRIC FORMULA:
C            GIVEN ANGLES B' and C' AND INCLUDED SIDE A,
C               1)  A'=180-B'-C',
C               2)  B=A*SIN(B')/SIN(A'),
C               3)  C=A*SIN(C')/SIN(A').
C          IF 'LS' AND/OR 'RS' ARE GREATER THAN ZERO, THE
C          BREAKPOINTS ARE (1,LS) AND/OR (256,RS).  'CRVGEN'
C          IS CALLED TO GENERATE THE CURVE, GIVEN THE
C          BREAKPOINTS.  THE HISTOGRAM 'HIST' IS THEN SCALED
C          TO THE CORRECT NUMBER OF POINTS USING 'NREC'.
C
C          SUBROUTINES USED - CRVGEN,INPUT
C
C
      REAL*4 HIST(256),ST(4),M,LS,RS,H
  100 CALL INPUT (ST)
      M=ST(1)
      LS=ST(2)
      RS=ST(3)
      H=ST(4)
      BB=H
      IF ((RS.NE.0).OR.(LS.NE.0).OR.(H.NE.0)) GO TO 150
      IF (M.EQ.0.5) GO TO 230
      IF (M.LT.0.5) GO TO 110
      FL=2.0*M-1.0
      FH=1.0
      GO TO 120
  110 FL=0.0
      FH=2.0*M
  120 DH=(NREC-2)*8.0/(FH-FL)
      IL=255.0*FL+1.5
      IH=255.0*FH+1.5
      DO 130 I=1,256
      HIST(I)=0.0
  130 CONTINUE
      DO 140 I=IL,IH
      HIST(I)=DH
  140 CONTINUE
      GO TO 200
  150 ANGINC=0.0174532925
      PIBY2=1.570796327
      IF (LS.GT.0) GO TO 160
      THETA=PIBY2+LS*ANGINC
      PHI=ATAN2(1.0,M)
      A=M*COS(THETA)*SIN(PHI)/SIN(THETA+PHI)
      AA=M*SIN(THETA)*SIN(PHI)/SIN(THETA+PHI)
      GO TO 170
  160 A=0.0
      AA=LS
```

```
  170 IF (RS.GT.0) GO TO 180
      ALPHA=PIBY2+RS*ANGINC
      BETA=ATAN2(1.0,(1.0-M))
      C=(1.0-M)*COS(ALPHA)*SIN(BETA)/SIN(ALPHA+BETA)
      C=1.0-C
      CC=(1.0-M)*SIN(ALPHA)*SIN(BETA)/SIN(ALPHA+BETA)
      GO TO 190
  180 C=1.0
      CC=RS
  190 IA=255.0*A+1.5
      IB=255.0*M+1.5
      IC=255.0*C+1.5
      CALL CRVGEN (HIST,IA,IB,IC,AA,BB,CC)
  200 TO=0.0
      DO 210 I=1,256
      TO=TO+HIST(I)
  210 CONTINUE
      FH=(NREC-2)*2048.0/TO
      DO 220 I=1,256
      HIST(I)=HIST(I)*FH
  220 CONTINUE
      GO TO 250
  230 FH=(NREC-2)*8.0
      DO 240 I=1,256
      HIST(I)=FH
  240 CONTINUE
  250 RETURN
      END


      SUBROUTINE READO (LUN,NREC,ERFLG,LPIC)
C
C
C        THIS SUBROUTINE READS A BLOCK OF DATA FROM RECORD
C        'NREC' OF FILE 'LUN'.  THE DATA IS BYTE DATA ON THE
C        RANGE (-128,+127).  THIS IS CONVERTED TO INTEGER
C        DATA ON THE RANGE (0,255) AND STORED IN 'LPIC'.
C        IF 'ERFLG'=0, THERE WAS NO ERROR.
C
C        SUBROUTINES USED - NONE
C
C
      INTEGER*2 ERFLG,J
      LOGICAL*1 LPIC(4096),T(2)
      EQUIVALENCE (J,T(1))
      ERFLG=0
      READ (LUN'NREC,END=110,ERR=120) (LPIC(I),I=1,4096,2)
      DO 100 I=1,4096,2
      T(2)=0
      T(1)=LPIC(I)
      IF (J.GE.128) J=J-256
      J=J+128
```

```
      LPIC(I+1)=0
      LPIC(I)=T(1)
  100 CONTINUE
      RETURN
  110 WRITE (7,1) LUN
      GO TO 130
  120 WRITE (7,2) LUN
  130 ERFLG=0
    1 FORMAT (' END OF FILE ON UNIT ',I3,' IN ''READO''')
    2 FORMAT (' I/O ERROR ON UNIT ',I3,' IN ''READO''')
      RETURN
      END


      SUBROUTINE RECNUM (LPIC,NREC,PICNAM,LUN,ERFLG)
C
C
C        THIS SUBROUTINE READS PART OF THE LABEL RECORD OF
C        THE PICTURE ON FILE 'LUN' INTO THE BUFFER 'LPIC'.
C        IT THEN ASCERTAINS THE SIZE OF THE PICTURE - S, M,
C        OR L, AND DETERMINES THE NUMBER OF RECORDS IN THE
C        FILE 'NREC'.  IT ALSO RETURNS THE NAME OF THE
C        PICTURE 'PICNAM'.  'ERFLG' INDICATES AN ERROR IF IT
C        IS NOT ZERO.
C
C        SUBROUTINES USED - NONE
C
C
      LOGICAL*1 LPIC(4096),PICNAM(8),S,M,L
      INTEGER*2 ERFLG,NREC
      DATA S,M,L/'S','M','L'/
      NREC=0
      ERFLG=0
      READ (LUN'1,END=110,ERR=120) (LPIC(I),I=1,2048)
      IF (LPIC(4).EQ.S) NREC=10
      IF (LPIC(4).EQ.M) NREC=34
      IF (LPIC(4).EQ.L) NREC=130
      IF (NREC.EQ.0) WRITE (7,1)
      DO 100 I=1,8
      PICNAM(I)=LPIC(I+8)
  100 CONTINUE
      RETURN
  110 WRITE (7,2) LUN
      GO TO 130
  120 WRITE (7,3) LUN
  130 ERFLG=1
    1 FORMAT (' INCORRECT PICTURE SIZE')
    2 FORMAT (' END OF FILE ON UNIT ',I3,' IN ''RECNUM''')
    3 FORMAT (' I/O ERROR ON UNIT ',I3,' IN ''RECNUM''')
      RETURN
      END
```

```
      SUBROUTINE RMSERR (REFHST,ACTHST,ERR)
C
C
C        THIS SUBROUTINE COMPUTES THE ROOT MEAN SQUARE
C        ERROR BETWEEN THE REFERENCE HISTOGRAM 'REFHST' AND
C        THE ACTUAL HISTOGRAM 'ACTHST'.  THE ERROR IS
C        COMPUTED WITH THE HISTOGRAMS SCALED SO THAT THE
C        SUMMATION OVER EACH HISTOGRAM IS ONE.
C
C        SUBROUTINES USED - NONE
C
C
      REAL *4 REFHST(256),ACTHST(256),ERR
      T=0.0
      DO 100 I=1,256
      T=T+REFHST(I)
  100 CONTINUE
      T1=0.0
      DO 110 I=1,256
      DELT=(REFHST(I)-ACTHST(I))/T
      T1=T1+DELT*DELT
  110 CONTINUE
      ERR=SQRT(T1/256.0)
      RETURN
      END


      SUBROUTINE RWLBLO (LUN1,LUN2,SYM,SYSDAT,LABEL,ERFLG)
C
C
C        THIS SUBROUTINE READS THE LABEL RECORD OFF OF THE
C        INPUT FILE 'LUN1', SUBSTITUTES THE APPROPRIATE
C        PROGRAM IDENTIFIER 'SYM' INTO THE PICTURE NAME,
C        AND WRITES THE LABEL RECORD INTO THE OUTPUT FILE
C        'LUN2'.  IT ALSO CLEARS THE FLAG THAT DENOTES THAT
C        THE HISTOGRAM OF THE PICTURE IS IN THE LABEL
C        RECORD.  IT UPDATES THE SYSTEM DATE WORD, AS
C        SUPPLIED BY 'SYSDAT'.  'LABEL' IS USED AS AN I/O
C        BUFFER.  'ERFLG' INDICATES THERE HAS BEEN AN ERROR
C        IF IT IS NOT ZERO.
C
C        SUBROUTINES USED - NONE
C
C
      LOGICAL*1 LABEL(4096),SYM,SYSDAT(2)
      READ (LUN1'1,END=100,ERR=110) (LABEL(I),I=1,2048)
      LABEL(16)=SYM
      LABEL(19)=SYSDAT(1)
      LABEL(20)=SYSDAT(2)
      LABEL (1024)=0
      WRITE (LUN2'1,END=130,ERR=120) (LABEL(I),I=1,2048)
```

```
      READ (LUN1'2,END=100,ERR=110) (LABEL(I),I=1,2048)
      WRITE (LUN2'2,END=130,ERR=120) (LABEL(I),I=1,2048)
      RETURN
  100 WRITE (7,1) LUN1
      GO TO 140
  110 WRITE (7,2) LUN1
      GO TO 140
  120 WRITE (7,2) LUN2
      GO TO 140
  130 WRITE (7,1) LUN2
  140 ERFLG=1
    1 FORMAT (' END OF FILE ON UNIT ',I3,' IN ''RWLBLO''')
    2 FORMAT (' I/O ERROR ON UNIT ',I3,' IN ''RWLBLO''')
      RETURN
      END


      SUBROUTINE STAT (ST,HIST)
C
C
C        THIS SUBROUTINE COMPUTES THE STATISTICS OF THE
C        PICTURE WHOSE HISTOGRAM IS STORED IN 'HIST'.  THE
C        STATISTICS COMPUTED INCLUDE THE MEAN, MODE, MEDIAN,
C        MINIMUM AND MAXIMUM VALUES PRESENT, AND
C        COEFFICIENTS OF SKEWNESS AND KURTOSIS.  THE VALUE
C        OF THE HISTOGRAM AT THE MODE IS ALSO COMPUTED.
C        THESE VALUES ARE STORED IN ARRAY 'ST' AS WELL AS
C        BEING OUTPUT TO THE TERMINAL DEVICE.
C
C        SUBROUTINES USED - NONE
C
C
      REAL*4 HIST(256),ST(11)
      REAL*8 T1,T2,T3,T4
      T0=0.0
      T1=0.0
      T2=0.0
      T3=0.0
      T4=0.0
      TM=0.0
      ST(7)=255.0
      ST(8)=0.0
      DO 100 I=1,256
      IF (HIST(I).EQ.0.0) GO TO 100
      T=I-1
      T0=T0+HIST(I)
      T1=T1+HIST(I)*T
      T2=T2+HIST(I)*T*T
      T3=T3+HIST(I)*T*T*T
      T4=T4+HIST(I)*T*T*T*T
      TM=AMAX1(TM,HIST(I))
```

```
      IF (TM.EQ.HIST(I)) ST(6)=T
      ST(8)=AMIN1(ST(8),T)
      ST(9)=AMAX1(ST(9),T)
  100 CONTINUE
      ST(5)=TM
      T1=T1/T0
      T2=T2/T0
      T3=T3/T0
      T4=T4/T0
      ST(1)=T1
      ST(2)=T2-T1*T1
      ST(3)=T3-3.0*T1*T2+2.0*T1*T1*T1
      ST(4)=T4-4.0*T1*T3+6.0*T1*T1*T2-3.0*T1*T1*T1*T1
      ST(10)=ST(3)/(ST(2)**1.5)
      ST(11)=(ST(4)/(ST(2)*ST(2)))-3.0
      T1=0.0
      TM=T0/2.0
      I1=ST(8)+1
      I2=ST(9)+1
      DO 110 I=I1,I2
      T=I-1
      T1=T1+HIST(I)
      IF (T1.GE.TM) GO TO 120
  110 CONTINUE
  120 ST(7)=T
      WRITE (7,1) ST
    1 FORMAT (' M1 = ',E11.4,' M2 = ',E11.4,' M3 = ',E11.4,
     1' M4 = ',E11.4,/,' MODE = ',F8.1,' AT ',F5.1,
     2' MEDIAN = ',F5.1,' MIN = ',F5.1,' MAX = ',F5.1,/,
     3' A3 = ',E11.4,' A4 = ',E11.4)
      RETURN
      END


      SUBROUTINE TRHIST (TAB,HIST1,HIST2)
C
C
C        THIS SUBROUTINE TRANSFORMS AN INPUT HISTOGRAM
C        'HIST1' INTO AN OUTPUT HISTOGRAM 'HIST2' USING THE
C        TRANSFORMATION 'TAB'.
C
C        SUBROUTINES USED - NONE
C
C
      INTEGER*2 TAB(256)
      REAL*4 HIST1(256),HIST2(256)
      DO 100 I=1,256
      HIST2(I)=0.0
  100 CONTINUE
      DO 110 I=1,256
      J=TAB(I)+1
```

```
      HIST2(J)=HIST2(J)+HIST1(I)
  110 CONTINUE
      RETURN
      END


      SUBROUTINE UNIRMS (NREC,HIST,ERR)
C
C
C        THIS SUBROUTINE COMPUTES THE ROOT MEAN SQUARE ERROR
C        BETWEEN 'HIST' AND A UNIFORM HISTOGRAM WITH THE
C        SAME NUMBER OF PIXELS AS DETERMINED BY 'NREC'.  THE
C        ERROR IS COMPUTED WITH THE HISTOGRAMS SCALED SO
C        THAT THEY SUM TO 1.  'ERR' IS THE RESULTING ERROR.
C
C        SUBROUTINES USED - NONE
C
C
      REAL*4 HIST(256),ERR
      PN=(NREC-2)*8.0
      PNH=(NREC-2)*2048.0
      T=0.0
      DO 100 I=1,256
      DELT=(PN-HIST(I))/PNH
      T=T+DELT*DELT
  100 CONTINUE
      ERR=SQRT(T/256.0)
      WRITE (7,1) ERR
    1 FORMAT (' RMS ERROR WITH RESPECT TO UNIFORM = ',
     1E11.4)
      RETURN
      END


      SUBROUTINE WRITEO (LUN,NREC,LPIC,ERFLG)
C
C
C        THE INPUT TO THIS SUBROUTINE IS AN ARRAY OF INTEGER
C        DATA IN 'LPIC' THAT HAS VALUES ON THE RANGE
C        (0,255).  THIS DATA IS THEN CONVERTED TO BYTE DATA
C        ON THE RANGE (-128,+127) AND WRITTEN TO RECORD
C        'NREC' IN FILE 'LUN'.  'ERFLG' INDICATES AN ERROR
C        HAS OCCURRED IF IT IS NOT ZERO.
C
C        SUBROUTINES USED - NONE
C
C
      INTEGER*2 J,ERFLG
      LOGICAL*1 LPIC(4096),T(2)
      EQUIVALENCE (J,T(1))
```

```
      ERFLG=0
      T(2)=0
      DO 100 I=1,4096.2
      T(1)=LPIC(I)
      J=J-128
      IF (J.LT.0) J=J+256
      LPIC(I)=T(1)
  100 CONTINUE
      WRITE (LUN'NREC,END=110,ERR=120) (LPIC(I),I=1,4096,2)
      RETURN
  110 WRITE (7,1) LUN
      GO TO 130
  120 WRITE (7,2) LUN
  130 ERFLG=1
    1 FORMAT (' END OF FILE ON UNIT ',I3,' IN ''WRITEO''')
    2 FORMAT (' I/O ERROR ON UNIT ',I3,' IN '' WRITEO''')
      RETURN
      END



      SUBROUTINE XLATE (NREC,LUN1,LUN2,ERFLG,IPIC,TABLE)
C
C
C        THIS SUBROUTINE READS THE DATA FROM THE INPUT FILE
C        'LUN1', TRANSFORMS IT ACCORDING TO 'TABLE', AND
C        WRITES IT INTO THE OUTPUT FILE 'LUN2'.  THIS IS
C        DONE FOR THE WHOLE PICTURE ACCORDING TO 'NREC'.
C        'IPIC' IS USED AS AN I/O BUFFER.  'ERFLG' IS THE
C        FLAG THAT SAYS THAT THERE WAS AN ERROR IF NOT ZERO.
C
C        SUBROUTINES USED - READO,WRITEO
C
C
      INTEGER*2 IPIC(2048),TABLE(256),ERFLG
      DO 110 I=3,NREC
      CALL READO (LUN1,I,ERFLG,IPIC)
      IF (ERFLG.NE.0) GO TO 999
      DO 100 J=1,2048
      M=IPIC(J)+1
      IPIC(J)=TABLE(M)
  100 CONTINUE
      CALL WRITEO (LUN2,I,IPIC,ERFLG)
      IF (ERFLG.NE.0) GO TO 999
  110 CONTINUE
  999 RETURN
      END
```

```
        .TITLE  CURSOR SAMPLER
; THIS SUBROUTINE SAMPLES THE JOYSTICK ON THE DIGITIZER
; AND RETURNS THE X, Y, AND Z COORDINATES OF THAT POINT.
; IT IS CALLED FROM FORTRAN IN THE FOLLOWING MANNER:
;       CALL CURSOR (IX,IY,IZ)
; IX IS THE X COORDINATE OF THE JOY STICK CURSOR
; IY IS THE Y COORDINATE OF THE JOY STICK CURSOR
; IZ IS THE Z VALUE AT (IX,IY) AND IS OPTIONAL
;
;
; DEFINITIONS
;
        CREG    =164120         ; CONTROL AND STATUS
        XREG    =164122         ; SAMPLE X
        YREG    =164124         ; SAMPLE Y
        ZREG    =164126         ; Z OUTPUT
        JOXREG  =164130         ; JOY STICK X
        JOYREG  =164132         ; JOY STICK Y
        .GLOBL  CURSOR
        .MCALL  .REGDEF,..V2..
        ..V2..
        .REGDEF
;
;
CURSOR: MOV     @#JOXREG,@2(R5) ; SAVE X
        MOV     @#JOYREG,@4(R5) ; SAVE Y
        MOV     @2(R5),@#XREG   ; SAMPLE X
        MOV     @4(R5),@#YREG   ; SAMPLE Y
        CMP     (R5),#2         ; SAMPLE?
        BEQ     OUT             ; NO
        MOV     #1,@#CREG       ; GO
LP1:    BIT     #200,@#CREG     ; WAIT
        BEQ     LP1
        MOV     @#ZREG,@6(R5)   ; GET Z
OUT:    RTS     PC
        .END


        .TITLE JOYSTK SIMULATOR
; THIS SUBROUTINE SIMULATES A JOYSTICK WITH FOUR DEGREES
; OF FREEDOM.  IT USES THE FOUR POTENTIOMETERS ON THE
; LABORATORY PERIPHERAL SYSTEM UNIT.  IT IS CALLED IN THE
; FOLLOWING MANNER:
;
;       CALL JOYSTK (IA,IB,IC,ID)
;
; WHERE IA, IB, IC AND IB ARE INTEGER*2 VARIABLE THAT
; REPRESENT THE FOUR PARAMETERS.  THE RANGE OF VALUES IS
; (-2048,+2047).
;
;
```

```
; WHEN THE SUBROUTINE IS CALLED, ALL OF THE LED'S ON THE
; LPS UNIT ARE BLANKED.  THEN THE ROUTINE GOES INTO A LOOP
; THAT OPERATES IN THE FOLLOWING MANNER:
;       1)  BEGIN WITH CHANNEL 0
;       2)  SAMPLE CHANNEL, OFFSET IT TO (-2048,+2047)
;       3)  MOVE BLANK OR MINUS SIGN INTO LED#2 AS REQUIRED
;       4)  COMPUTE INTEGER PERCENT (-99,+99)
;       5)  OUTPUT DIGITS TO LED'S 0 AND 1
;       6)  LOOK FOR INPUT FROM TTY -- IF NONE, GO TO 2
;       7)  SAMPLE CHANNEL, STORE IN CORRECT LOCATION
;       8)  INCREMENT CHANNEL # -- IF NOT FINISHED, GO TO 2
;       9)  EXIT
;
;
; DEFINITIONS
;
JSW     =44             ; RT-11 JOB STATUS WORD
ADS     =170400         ; LPS A-D STATUS REGISTER
ADB     =170402         ; LPS A-D BUFFER REGISTER
MINUS   =1015           ; MINUS SIGN FOR LED#2
PLUS    =1017           ; BLANK IN LED #2
        .GLOBL  JOYSTK
        .MCALL  ..V2..,.REGDEF,.TTINR,.TTYOUT
        ..V2..
        .REGDEF
;
;
JOYSTK: CLR     @#ADS           ; CLEAR STATUS REGISTER
        MOV     #17,R1          ; CODE FOR BLANK TO LED 0
        MOV     #6,R2           ; SIX LED'S
1$:     MOV     R1,@#ADB        ; OUTPUT BLANK
        ADD     #400,R1         ; INCREMENT LED NUMBER
        SOB     R2,1$           ; LOOP
        TST     (R5)+           ; BUMP ARGUMENT POINTER
        BIS     #10100,@#JSW    ; SET TTY BITS IN JSW
        MOV     #4,R4           ; FOUR CHANNELS TO SAMPLE
2$:     INC     @#ADS           ; START CONVERSION
        TSTB    @#ADS           ; WAIT FOR READY
        BPL     .-4
        MOV     @#ADB,R2        ; MOVE VALUE TO R2
        SUB     #4000,R2        ; OFFSET TO (-2048,+2047)
        MOV     #PLUS,@#ADB     ; MOVE BLANK TO LED #2
        TST     R2              ; IS R2 NEGATIVE?
        BPL     3$              ; NO
        MOV     #MINUS,@#ADB    ; YES -- PUT OUT MINUS SIGN
        NEG     R2              ; TAKE ABSOLUTE VALUE
;
; THIS SECTION OF CODE CONVERTS THE CONTENTS OF R2 WHICH IS
; ON THE INTERVAL (-2048,+2047) TO A PERCENTAGE (-99,+99).
; IT ROUNDS OFF THE RESULT RATHER THAN TRUNCATE IT.
; A PERCENTAGE IS USUALLY COMPUTED BY R2*100/2048.  THIS
; SUBROUTINE INSTEAD MULTIPLIES BY 3200 AND THEN TAKES THE
```

```
; HIGH ORDER 16 BITS OF THE ANSWER.  THIS ALLEVIATES THE
; NECESSITY FOR A DIVIDE.
;
3$:     MUL     #3200.,R2       ; MULTIPLY BY 3200
        ROL     R3              ; GET MSB OF LOW 16 BITS
        ADC     R2              ; ROUND OFF ANSWER
        CMP     R2,#100.        ; COMPARE TO 100
        BLT     4$              ; OK
        MOV     #99.,R2         ; SET TO 99 IF GREATER
4$:     MOV     R2,R1           ; MOV TO LOW 16 BITS OF 32
        CLR     R0              ; CLEAR HIGH 16 BITS
        DIV     #10.,R0         ; CONVERT TO DECIMAL DIGITS
        ADD     #400,R0         ; PUT HI ORDER IN LED #1
        MOV     R0,@#ADB        ; OUTPUT LED #1
        MOV     R1,@#ADB        ; OUTPUT LED #0
        .TTINR                  ; LOOK FOR TTY INPUT
        BCS     2$              ; BRANCH IF NONE
        .TTYOUT                 ; OUTPUT WHAT WAS INPUT
        INC     @#ADS           ; SAMPLE SAME CHANNEL
        TSTB    @#ADS           ; WAIT ON READY
        BPL     .-4
        MOV     @#ADB,@(R5)     ; STORE VALUE
        SUB     #4000,@(R5)+    ; OFFSET VALUE
        INCB    @#ADS+1         ; INCREMENT CHANNEL NUMBER
        SOB     R4,2$           ; LOOP
        .TTYOUT #15             ; OUTPUT CARRIAGE RETURN
        BIC     #10100,@#JSW    ; RESET JSW
        RTS     PC              ; RETURN
        .END
```